PRIJS ƒ 7.25/Bfr. 135

Commodore INFO

Flightsimulators

Robotbasic

Ocean Software

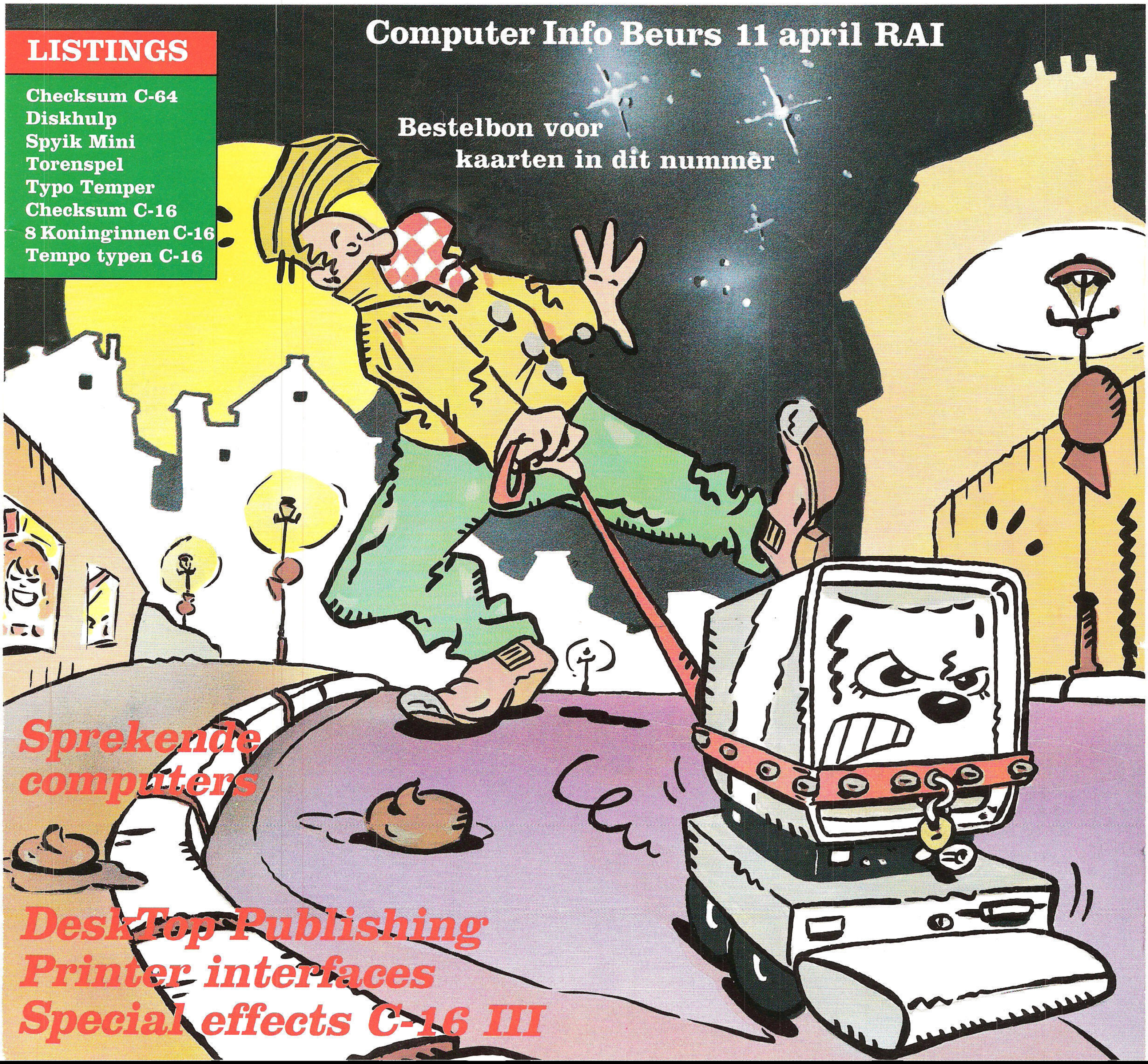
ONAFHANKELIJK BLAD VOOR COMMODORE GEBRUIKERS JAARGANG 4, NO. 2, MRT./APRIL 1987

Computer Info Beurs 11 april RAI

LISTINGS

- Checksum C-64
- Diskhulp
- Spyik Mini
- Torenspel
- Typo Temper
- Checksum C-16
- 8 Koninginnen C-16
- Tempo typen C-16

Bestelbon voor kaarten in dit nummer



Sprekende computers

DeskTop Publishing

Printer interfaces

Special effects C-16 III

COMMODORE-INFO

Jaargang 4, no. 2, mrt./april 1987

Uitgave: Sala Communications

Uitgever: Drs. J. Taverne

REDAKTIE

Ir. L. Sala hoofdredacteur
J. Bodzinga adj. hoofdred.
K. van der Vlies reportage
B. Munniksma onderwijs
Sj. Bakker machinetaal
U. Schuurmans software
R. Goudriaan listings

ART DEPARTMENT

Strip: Bert Tier

Illustraties: Ben van Mierlo
Ymmot

Advertentie-exploitatie:
Ing. V. Sala
R. Akker

Redaktieadres: Postbus 112
1260 AC Blaricum
☎ 02152-65695

Abonnementen en administratie: Postbus 5570
1007 AN Amsterdam
Den Texstraat 5a
1017 XW Amsterdam
☎ 020-273198

Vragen betreffende abonnementen bij voorkeur schriftelijk, met meesturen van het omslagetiket. Telefonisch uitsluitend van 10.00 tot 15.00 uur.

Abonnement: (8 nummers)
ƒ 47,50 of Bfr. 950 per jaar

Betaling op Giro 1585491 tnv. SAC/COMMODORE-INFO Blaricum of in België op Bank BBL nr. 310050602562, vermeld SAC/COMMODORE-INFO. Oude nummers à ƒ 6,75 alleen bij vooruitbetaling op een van bovenstaande rekeningen. Ook telefonische opgave voor een abonnement is mogelijk. Bel GRATIS 06-0224222, HP-Teleservice, elke dag tot 20.30 uur (ook in het weekend). Dit nummer is alleen voor telefonische opgave van NIEUWE abonnementen.

Redactiesecretariaat:
Ron van Zalinge

Druk: Verweij, Mijdrecht
NDB, Zoeterwoude

Distributie:
in Nederland Betapress/Gilze
in België AMP/Brussel

ISSN: 0169-3085

Inhoud van dit nummer



REDACTIONEEL

Toen we in de VS in Januari op de CES in Las Vegas de nieuwe Amiga's van Commodore zagen, waren we natuurlijk verrast. We hadden grote verhalen in de pen. Maar terug in Nederland verzocht de directie van Commodore ons hier niet over te schrijven tot na de officiële introductie, 7 maart in Hannover. Hetgeen we dan ook niet gedaan hebben, als redactie houden we ons aan dergelijke afspraken. De rest van de journalistiek schijnt daar minder moeite mee te hebben, de tests vliegen je om de oren. Of Commodore nu weer eens amateur media-politiek bedrijft, of gewoon geen goede afspraken maakt, weten we niet. In ieder geval in Commodore-Info nog geen officieel nieuws over de nieuwe 500 en 2000.
Zoals gebruikelijk zal het ook nog wel even duren voordat deze CBM machines echt te koop zijn, dus we komen er echt nog op tijd op terug.
We hopen u de nieuwe Amiga's in ieder geval te kunnen laten zien op 11 April in de RAI. Kom daar ook zeker kijken naar de DeskTop Publishing Shootout. Daar gaan we de verschilende DeskTop pakketten testen en u kunt daar als publiek getuige van zijn.

L.S.

Gebruikersgroepen

De **VCGN** (070-971851) is de oudste Commodore gebruikersvereniging en heeft iedere laatste zaterdag van de maand in het Lodewijk Makeblijde lyceum in Rijswijk een grote bijeenkomst. Inlichtingen over de vele afdelingen kunt u krijgen via het secretariaat: Haagweg 155, 2281 AH Rijswijk, 070-950779.
Verder afdelingen: Zeeland 01184-14789, West-Brabant 01640-42404, Midden-Brabant 01612-4242, Zuid-Limburg 04490-19946, Zuid-Holland 078-135739.

PET Benelux heeft bijeenkomsten in Zeist (1e zaterdag), Nijmegen (2e zaterdag) en Haaksbergen (3e zaterdag). (Inl.05759-2211).

De **HCC** Commodore gebruikersgroep heet Compet, 03404-59599.

HOT NEWS van SCN uit Amsterdam (03450-16051) heeft haar maandelijkse bijeenkomst in de Meervaart op de 2e zaterdag van de maand en ook activiteiten in Zwolle (038-548459), Midden Nederland (03453-1375) en Deventer (05700-50758).

Voor **C-128** gebruikers is er een initiatief voor een GG, inl. 02997-1479, Zuideinde 1, 1843 JL Grootschermer.

In **Arnhem** zit een Commodore-club, waar men clubavonden op dinsdag iedere twee weken organiseert en ook cursussen geeft. Inl. Hofsingel 259 te 6834 GH Velp, tel. 085-647782.

De Dutch PC Club organiseert iedere 2 zaterdag eeen bijeenkomst voor alle merken in de Kajuit, Hazekampseweg 36 in **Nijmegen**, Inl. Spechtstr. 32, 4043 LS Opheusden.

Den Helder heeft de Commodore CG, te bereiken via Postbus 2068, 1780 BC te Den Helder of telefonisch via 02230-31430.

In **Groningen** is er elke 2e zaterdag van de maand een gebruikersbijeenkomst in de Beijumkorf, Jaltadadreff te Groningen, aanvang 10.30 u.

In **Noordwijk** zit een C-64/128 gebruikersgroep, die iedere 1e dinsdag een bijeenkomst heeft. Inlichtingen A. vd Klaauw ☎ 01719-16720.

Oss heeft de vereniging ACC, voor Oost-Brabant en de Betuwe, via postbus 215 of telefoon 04120-42054/37649 bereikbaar. Bijeenkomsten iedere derde zaterdag in "de Pas" te Heesch.

In **Uden** zit de CCU, met elke eerste donderdag om 20.00 u een bijeenkomst in de schouw. 04132-64239

In **Weert** komt de Computer Club Weert iedere eerste en derde woensdag van de maand vanaf 20.00 uur bij elkaar in Hotel/ Pub de Rogsteker, Langstraat 15, Weert. Meer informatie bij P. Schijven. 04956-3146

Voor de **Amiga** is er ook een club, te breiken via Postbus 41138, 1009 EC te Amsterdam. Inl. 020-931191.

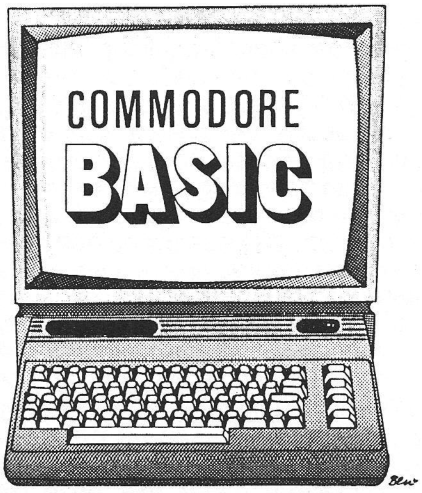
BELGIE
De **CCCB** is een van de clubs uit Belgie, met afdelingen in Antwerpen, Limburg, West-Vlaanderen inl. Bindstr. 19, te 2600 Berchem Antwerpen. Lid worden door storting 500 fr op 405-5092171-90 tnv. CCCBA.
Ook de **HCC België** heeft bijeenkomsten iedere 2e en 4e maandag op de Ottergemse steenweg-Zuid 705 te Gent (03-6468961).

In **Oostende** is er de Commodore-Club Oostende, Raversijdestraat 48, B 8400 Oostende 059-805191.
In **Borgerhout** is er een C-64 gebruikersgroep, die de 2e en 4e vrijdag van de maand om 19.30 u bijeenkomt in "de Sportschuur". Voor inlichtingen 03-2710718. ●

In deze serie artikelen nemen we byte voor byte de Basic-interpreter van de Commodore-computers onder de loep. We houden ons daarbij vooral bezig met de structuur van de Basic programma's, zonder direct te maken te krijgen met ingewikkelde PEEKs en POKEs. Daardoor kan deze cursus niet alleen door C-64 bezitters worden gebruikt, maar ook de Amiga- en C-16 lezers komen hier aan hun trekken. Zelfs de oude business machines van Commodore kunnen met deze Basic uit de voeten. Jan Bodzinga gaat in deze aflevering in op een ingewikkelde Basic-techniek: modulaire programma's en het werken met meer-dimensionale arrays.

Basis Basic

Deel 12
Een Database (I)



Dat het programmeren in Basic niet alleen leuk kan zijn, maar ook nog erg nuttig, hopen we in deze en de komende afleveringen van onze cursus wat nader te bekijken. Tot nog toe hebben we ons voornamelijk met losse stukken theorie bemoeid, waarbij we zo nu en dan een uitstapje maakten naar een compleet Basic-programma. Deze les maakt een begin met het ontwerpen van een echte, zelfgeschreven, database.
We zullen de ontwikkeling van het programma vanaf het begin volgen, waarbij we in meerdere afleveringen uiteindelijk terecht komen bij een menu-gestuurd programma waarmee we professioneel een adressenbestand kunnen opzetten en bijhouden.

Fase 1
De eerste fase van de ontwikkeling van een software programma is altijd het ontwerp. Om een goed stuk programmatuur te schrijven is het nodig vooraf te weten wat er precies moet gebeuren. Deze fase wordt door de beginnende programmeur meestal lichtvoetig over geslagen, met alle gevolgen van dien. Het uiteindelijke resultaat ziet er dan ook vaak uit als een bord spaghetti, waarbij de GOTO's de slierten software aan elkaar proberen te breien. Al met al geven deze Basic-kunsten meestal niet het gewenste resultaat. Het is dan ook niet onverstandig, om wat nader stil te staan bij deze ontwerpfase.
Als eerste moet je daarbij de vraag stellen, wat het programma precies moet gaan doen. Daarna moet worden bekeken in hoeverre de beschikbare mogelijkheden van hard- en software het programma mogelijk maken, en wat daarbij de beperkingen zijn. Het zal bijvoorbeeld voor iedereen duidelijk zijn, dat het niet verstandig is, om een adressenbestand van zo'n 10.000 entries (ingevoerde adressen) op een Commodore 64 te gaan schrijven. Al was het maar, omdat de diskdrive van de 64/128 te weinig opslagcapaciteit heeft om zoveel gegevens efficiënt te kunnen bevatten.
We moeten dus in alle gevallen roeien met de riemen die we beschikbaar hebben. Voor ons betekent dat, dat er sprake is van beperkte mogelijkheden, maar ons ontwerp zal daarentegen wel model kunnen staan voor een programma dat (later) ontwikkeld kan worden op een grotere machine zoals een PC of zelfs een mainframe. Dus ook degenen die inmiddels al dromen van het grotere werk, kunnen hun voordeel halen uit de komende afleveringen van deze cursus.

Data-bestanden
Voor het maken van een goed gegevensbestand moeten we ons eerst afvragen, wat we nu werkelijk als doel hebben. Op zichzelf lijkt dat wellicht eenvoudiger dan het is. Want er gaat nogal wat om in zo'n bestand. Vaak beseft men nauwelijks wat er allemaal komt kijken bij het verwerken van een data-bestand. Daarbij maakt het eigenlijk niets uit of het nu gaat om een bestand met adressen en telefoonnummers, een database met titels en componisten van grammofoonplaten of de gegevens van alle 10.000 kippen in een legbatterij. En er zijn nog talloze voorbeelden te bedenken van **gegevens-verzamelingen** die passen in een database.

We kunnen een bestand van deze orde eigenlijk vergelijken met de ouderwetse **kaartenbak**. Voor ieder onderdeel van het bestand, adres, grammofoonplaat, kip of keuken-

recept, zit in de kaartenbak een eigen kaart. Komt er een gegeven bij, dan wordt een nieuwe kaart in de bak gestoken. Meestal gebeurt dit handmatig en netjes op alfabetische volgorde.
Wordt er in de kaartenbak een onderdeel veranderd, dan worden op de kaart de oude gegevens doorgestreept en vervangen door de nieuwe. Als het gegeven in z'n geheel ongeldig is geworden, dan kan de kaart worden vernietigd.
In principe zijn deze drie zaken de belangrijkste bewerkingen van een gegevens-bestand:

- Gegevens toevoegen
- Gegevens wijzigen
- Gegevens verwijderen

Bij het werken met de computer als bestandsbeheerder, gelden deze drie bewerkingen natuurlijk ook. Als je 't goed bekijkt, zitten er eigenlijk maar heel weinig verschillen tussen de ouderwetse kaartenbak en de moderne automatiserings-technieken. Alleen de namen zijn hier en daar wat veranderd. Wat eertijds op **een kaart** werd geschreven noemen we nu een **record**. De afzonderlijke gegevens op een kaart, zoals naam, straat, woonplaats, noemen we nu de **velden** van dat record. De hele kaartenbak is in de computertaal omgedoopt tot gegevens-bestand, of **database**. Hier zijn nog wel wat varianten op te bedenken, maar met deze drie dimensies kunnen we uit de voeten.
Als we bovenstaande gegevens toepassen op een eenvoudig (adressen)bestand komen we tot het volgende schema:

```
DATABASE      (compleet bestand)
  RECORD 1    (1 adres)
    VELD1     (naam)
    VELD2     (straat)
    VELD3     (woonplaats)
  RECORD 2    (1 adres)
    VELD1     (naam)
    VELD2     (straat)
    VELD3     (woonplaats)
  RECORD 3    (1 adres)
    VELD1     (naam)
    VELD2     (straat)
    VELD3     (woonplaats)
  RECORD 4    (1 adres)
    VELD1     (naam)
    VELD2     (straat)
    VELD3     (woonplaats)
EINDE DATABASE
```

De bewerkingen op dit bestand hebben te maken met deze drie dimensies. Zelfs zonder gegevens kan er sprake zijn van een (hypothetische) database, hoewel onlogisch, is dat toch gebruikelijk in computerland. De definitie van een gegevensbestand zou dan ook kunnen luiden:

Een gegevensbestand of database is een verzameling van nul of meer records, ingedeeld in velden.

Hier hoef je niet van te schrikken, want meestal zitten er meer dan 0 records in de database, anders hoeven we er niet mee te gaan werken. Een record is echter geen record als er geen velden in zitten. Het verschil tussen die twee zit in het feit, dat er al sprake is van een bestand, als het een structuur heeft gekregen van de programmeur. In het voorgaande schema bestaat de structuur uit drie velden, *naam, adres en woonplaats*. Een record zonder velden kan eenvoudig **niet** bestaan, terwijl een database met structuur, maar (nog) zonder gegevens wel degelijk bestaansrecht heeft.

Ontwerp

Laten we de theorie op dit moment maar verwisselen voor de praktijk. Ons doel is om een Basic-programma op te zetten, waarmee we een professionele database kunnen onderhouden. Zoals al opgemerkt, betekent dit, dat we ons eerst zullen moeten afvragen welke structuur ons bestand zal moeten hebben, hoeveel velden er in één record moeten, en welke bewerkingen we met het bestand willen uitvoeren.

Data-structuur

Eerst de structuur van het bestand. Het programma dat we zullen gaan schrijven in Basic, beperkt zich tot 6 velden binnen een record. We hebben daarvoor gekozen, om het geheel niet al te ingewikkeld te maken, maar je zult zien, dat het uiteindelijk erg eenvoudig is, om de structuur van de database (en het programma) aan eigen wensen aan te passen. De velden van een record in ons bestand zouden we aan kunnen duiden met een nummer, dus veld 1, veld 2, veld 3 enzovoort. In de praktijk werkt dit niet erg gemakkelijk, want je weet op die manier nooit welk gegeven er precies in welk veld thuishoort. Basic is ons hierbij behulpzaam, door ons mogelijkheden te geven om de velden een **alfanumerieke** naam toe te kennen.
De **namen** die we daarvoor hebben gekozen hebben natuurlijk betrekking op de **inhoud** van dat veld.
Een record in ons ontwerp-programma heeft als veldindeling:

- veld 0 NAAM
- veld 1 VOORLETTERS
- veld 2 STRAAT EN NUMMER
- veld 3 POSTKODE
- veld 4 WOONPLAATS
- veld 5 TELEFOON

Een eenvoudig te begrijpen structuur, waarbij alleen het feit dat we bij veld 0 beginnen te tellen misschien een beetje vreemd is. Maar dat heeft te maken met de manier waarop we straks het programma opbouwen en laten runnen. We komen daar later nog wel op terug.

Bewerkingen

We hebben al eerder iets over de diverse bewerkingen gezegd, die we met een database kunnen uitvoeren. De voornaamste daarvan zijn het *toevoegen, muteren* en *verwijderen* van **veld-gegevens** en het complete **record**. Met de ouderwetse kaartenbak kwamen we ook niet erg veel verder dan deze drie. De overige bewerkingen kunnen we eigenlijk alleen met hulp van een computer (en printer) uitvoeren. We zullen er een paar op een rijtje zetten:

- Bestand wegschrijven.
- Bestand inlezen.
- Bestand sorteren op kenmerk.
- Record in bestand zoeken.
- Bestand printen.

Samen met de eerste drie bewerkingen komen we zo op een totaal van 8 manieren om met onze gegevens om te gaan. Genoeg dus, om een paar bladzijden Basic voor te kunnen bedenken.
Het ontwerp van onze programmastructuur bestaat hierdoor uit het opzetten van 8 grote subroutines, die elk één bewerking voor hun rekening nemen. Daarnaast hebben we natuurlijk te maken met de **'programma-overhead'**, waaronder we de initialisatie, het inlezen van DATA, de communicatie met de gebruiker en het beëindigen van het programma verstaan. Daarmee is het basic-ontwerp van ons programma klaar.
We kunnen nu overgaan tot het op papier zetten van de listing-structuur.

Programma-structuur

Omdat er nogal wat verschillende subroutines in ons programma voorkomen, is het verstandig een layout

te maken van de manier, waarop de listing kan worden opgezet. We kunnen dan aan de hand van de programmaregel-indeling de verschillende subroutines schrijven. Later kan er natuurlijk best van deze initiële indeling worden afgeweken, maar op dit moment is het goed om een listing schema in elkaar te zetten:

0000 - 0100	**REM-programmatitel**
0100 - 0300	**INITIALISATIE**
0300 - 1000	**Diverse subroutines**
1000 - 2000	**Programma-start(menu)**
2000 - 3000	**Toevoegen**
3000 - 4000	**Muteren**
4000 - 5000	**Verwijderen**
5000 - 6000	**Printen**
6000 - 7000	**Sorteren**
7000 - 8000	**Zoeken**
8000 - 9000	**Inlezen**
8000 - 9000	**Wegschrijven**
9500 - 9999	**Stoppen**

Elk van deze onderdelen kunnen we beschouwen als aparte **modules** in het programma, die uiteindelijk tot één listing worden samengevoegd. In principe zou het mogelijk zijn, om voor alle modules (bewerkingen) een apart programma te schrijven en te runnen. Dit is echter een tijdrovende en onhandige bezigheid. Het is veel beter, om al deze modules in één programma onder te brengen, waarbij met hulp van een **keuzemenu** door de gebruiker kan worden aangegeven welke bewerking er moet worden uitgevoerd. In onze structuur vind je dan ook vanaf regel 1000 de programma-start, waarbij het **menu** naar voren komt.
Laten we ons maar eens achter het toetsenbord begeven en deze startregels alvast intypen:

```
1000 REM begin programma
1010 SU=0:KZ=0:GOSUB 600 : REM
     menu
1020 IF KZ = TK THEN 9500 : REM
     einde
1030 ON KZ GOSUB 2000,3000,4000,
     5000,6000,7000,8000,9000
1040 GOTO 1010 : REM menu
```

Toegegeven, je kunt nog niet zoveel beginnen met dit stukje listing, maar het begin is er toch. Over deze regels, die het hart van de menubesturing vormen, is weinig te zeggen. Hierin vind je de enige **GOTO** die het programma rijk is, en deze dient, om na het uitvoeren van een bepaalde module weer terug te kunnen keren naar het begin van de menu-optie. Verder treffen we hier de **ON..GOSUB** functie aan, waarbij aan de hand van de waarde van variabele **KZ** naar een subroutine wordt gesprongen, die begint op één van de in regel 1030 vermelde regelnummers. Als **KZ** een waarde heeft van 1 gaat het programma verder op regel 2000, terwijl bij KZ=4 de subroutine vanaf regel 5000 zal worden uitgevoerd.
Het printwerk voor dit keuzemenu zit kennelijk vanaf regel 600 in het programma, afgaande op regel 1010.

Programma-begin

De eerste regels van onze database, dus alles wat er met de initialisatie te maken heeft, vinden we volgens ons schema voor regel 1000. Daar hebben we ook alle kleinere **subroutines** ondergebracht. De structuur is zodanig, dat vanaf regel 2000 iedere 1000 regels een nieuwe module (subroutine) begint, die met één van de keuzes in het menu te maken heeft.
Maar nu de allereerste regels van het programma:

```
10   REM DATABASE met modules
20   REM voor adressenbestand
30   REM Commodore-Info 2/1987
40   REM 870228/ v.1.02 J.B.
100  REM Initialisatie
110  REM
120  TT=0 : REM totaalaantal records
130  DIM A$(300,5) : REM aantal
     records max.
140  KZ=0 : REM keuze
150  I= 0 : SU=0: II = 0 :REM temp.var
160  DIM KZ$(10) : REM keuzestrings
170  DIM ST((LOG(301)/LOG(2)+4) , 1)
     :REM TEMP-ARRAYS
180  DIM VN$(7) : REM veldnamen
```

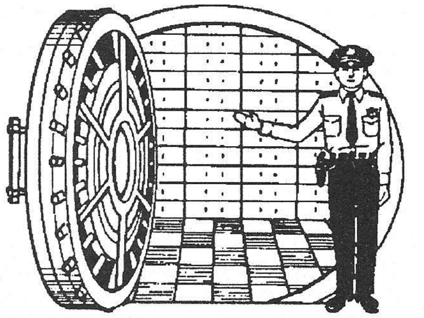
Na de gebruikelijke meldingen van versie, programmanaam en datum, die overigens ook belangrijk genoeg zijn om in ieder Basicprogramma te vermelden, vinden we vanaf regel 100 een opsomming van de belangrijkste variabelen, die een rol spelen in het hele programma. De variabele **TT** bijvoorbeeld geeft door alle modules heen het totaal aantal aanwezige records in het bestand. **KZ** houdt de in het menu gemaakte keuze vast en **SU, II** en **I** zijn een paar gewichtige tijdelijke variabelen, die als vlag of teller voor allerhande doeleinden worden gebruikt.

De **string/arrayvariabelen**, die in ons programma worden gebruikt zijn in feite nog belangrijker:
A$(300,5)
KZ$(10)
VN$(7)
ST((LOG(301)/LOG(2)+4),1)

De array **A$** gebruiken we voor de records, **KZ$** voor de omschrijving van de gemaakte keuze en **VN$** voor het onthouden van de veldnamen. **ST** is een numerieke array, waar we verder niet op in zullen gaan, omdat het in het licht van dit programma te ver voert. Deze laatste wordt gebruikt bij de toe te passen QUICK-SORT, die we in dit programma hebben geïntegreerd. Voor dit moment is de **array A$(300,5)** voor ons het bekijken waard.

Meer dimensies

Zoals te zien in het listingdeel waarin de DIM-regels zijn neergezet, vinden we niet alleen de bekende **DIM opdrachten** met een *variabele-naam* en tussen haakjes één getal, maar ook de array **A$(300,5).** Dit betekent, dat we voor **A$** in het RAM geheugen van de computer een pointerruimte reserveren, die niet alleen groot genoeg is om 301 strings in de array **A$** te bevatten, maar daarnaast per **index** ook nog eens 6 aparte strings. We noemen dit een array met twee *dimensies.* Als je er nog nooit mee te maken hebt gehad, zal dit op het eerste gezicht wat moeilijk lijken. In het gebruik is het werken met dergelijke arrays - mits goed toegepast- alleen maar erg handig.



Wat is hier het geval ?
De adressen, die we straks met dit programma willen invoeren en bewerken, komen als **records** in de array A$ terecht. Voor iedere index, van 0 tot 300 past er dus 1 adres in de array. Maar ieder adres heeft een aantal velden in het record. Deze velden zijn allemaal een onderdeel van dat record, dus ook van de **array A$**. We zouden als alternatief namelijk 6 aparte één-dimensionale arrays kunnen definiëren, voor ieder veld één, die dan door het index-nummer het verband tussen de diverse velden in een bepaald record aangeven. Dat geeft bij een goede structuur echter nogal wat moeilijkheden en veel overbodig geschrijf. Denk maar eens aan het sorteren van een ge-

gevens-bestand, dat op een dergelijke manier is opgezet. Daarom hebben we gekozen voor een *meer-dimensionale* array, waarbij voor ieder record de mogelijkheid bestaat 6 aparte velden in te voeren. In ons voorbeeld is er dus sprake van een DIM, waarbij in wezen **301*6 = 1806** string-pointers worden gereserveerd. Wel een echte geheugenvreter dus.
In record 3 van de nog aan te maken database hebben we dus een array-variabele met de volgende inhoud:

```
A$(2,0) = NAAM
A$(2,1) = VOORLETTERS
A$(2,2) = STRAAT EN NUMMER
A$(2,3) = POSTKODE
A$(2,4) = WOONPLAATS
A$(2,5) = TELEFOON
```

Je ziet meteen al het gemak door de complexiteit heen. Het is op deze manier erg simpel geworden, om één compleet record uit te printen met hulp van een FOR..TO..NEXT lus:

```
X    FOR II = 0 TO 5
XX   PRINT A$(2,II)
XXX  NEXT II
```

Terwijl ook de manier om alleen de postkode van het complete bestand uit te printen niet meer ruimte in beslag neemt dan drie Basic-opdrachten:

```
X    FOR II = 0 TO TT
XX   PRINT A$(II,3)
XXX  NEXT II
```

Duidelijker dan deze voorbeelden kan het nut en gemak van de meerdimensionale arrays haast niet worden geïllustreerd. Bedenk maar eens een slimme loop, waarin hetzelfde kan worden verwerkt, zonder daarbij gebruik te maken van meer-dimensionale arrays. Tien tegen één, dat het resultaat heel wat ondoorzichtiger is, dan bovenstaande regels.

Perspectieven

De theorie achter de 2-,3-,4- en nog meer dimensies tellende arrays heeft (uiteraard) te maken met de natuurlijke dimensies die we in allerhande zaken om ons heen tegenkomen. Zo kunnen we een enkelvoudige (standaard) array voorstellen als één tabel, de twee-dimensie array als een tabel met kolommen, -zowel naar beneden als naar rechts dus-, terwijl een array met drie dimensies daarnaast ook nog eens de diepte in gaat. Daarmee zouden we bijvoorbeeld in hetzelfde programma meerdere adressenbestanden tegelijkertijd kunnen bewerken.

De **vierde dimensie** is, net als in de realiteit om ons heen, ook bij de programma-technieken niet meer voor te stellen door iets vergelijkbaars. We gaan er daarom ook maar niet op in, temeer omdat er in de

> **Het is altijd verstandig, om te werken met een twee-dimensionale array, als er sprake is van records en velden.**

praktijk bijzonder weinig gebruik van wordt gemaakt. Om maar te zwijgen van de 255ste dimensie, die bij de Commodore computers ook nog haalbaar is in de array-sfeer. Practisch bekeken stelt dit natuurlijk niets meer voor, want bij onze dierbare Commodore Basic ontbreekt simpelweg het nodige RAM-werk om fatsoenlijk uit de voeten te kunnen met meer dan 3 dimensies bij arrays.

Initialisatie

Het begin van onze database start op regel 210 waar de eerste opdracht, een **GOSUB** wordt uitgevoerd :

```
200  REM systeem-initialisatie
210  GOSUB 900 : REM keuzedata
     lezen
299  GOTO 1000 : REM begin
     programma
```

Vanaf regel 900 worden de diverse vaste systeemgegevens ingelezen in de stabiele arrays via DATA en READ opdrachten. Waarbij de volgende programma-delen worden aangesproken :

```
899  REM inlezen keuzemogelijkheden
900  I = 0 : TK=0
910  READ KZ$(I)
920  IF KZ$(I) = "eind" THEN TK = I-1 :
     GOTO 950
930  I = I + 1
940  GOTO 910
950  FOR I = 0 TO 5
960  READ VN$(I)
970  NEXT I
```

De variabele **TK** staat voor het totaal-aantal keuzemogelijkheden. Dit is opzettelijk geen konstante waarde, omdat op deze variabele manier erg gemakkelijk extra keuzes aan de KZ$( )-array kunnen worden toegevoegd. Vandaar ook de wat vreemde test op **"eind"** in regel 920. Bij wijzigingen moet je er wel om denken de DIM-opdracht van KZ$( ) in het begin van het programma, aan te passen.
Het aantal veldnamen in VN$( ) is bewust konstant op 6 gezet, om het gehele programma niet al te ingewikkeld te maken. Hoewel VN$(5) de hoogste index in de array is, zijn er in totaal toch 6 veldnamen. Dit komt natuurlijk, omdat ook VN$(0) mee telt. Een denk- c.q. programmeer-fout wordt hier nogal snel mee gemaakt, dus denk bij het zelf programmeren wel om deze schijnbare tegenspraak.

Data

De data die bij de hiervoor beschreven routine hoort vinden we in het volgende programma gedeelte:

```
399  REM data keuzemogelijkheden
400  DATA "KEUZEMENU
     ADRESSENBESTAND"
410  DATA "Records toevoegen"
411  DATA "Records wijzigen"
412  DATA "Records verwijderen"
413  DATA "Records opzoeken"
414  DATA "Bestand sorteren"
415  DATA "Bestand printen"
416  DATA "Bestand inlezen"
417  DATA "Bestand wegschrijven"
418  DATA "Stoppen"
419  DATA "eind"
420  DATA "Naam              "
421  DATA "Voorletters       "
422  DATA "Straat en nummer  "
423  DATA "Postkode          "
424  DATA "Woonplaats        "
425  DATA "Telefoon          "
```

Tot zover de uitleg van ons programma. De volgende keer zullen we verder gaan met de andere routines en het complete menu. Helaas ontbreekt de ruimte om het hele programma met kanttekeningen in één keer te bespreken. Ik vind het jammer, dat we nu al moeten stoppen, maar dat zijn nu eenmaal zaken, waar je als redacteur mee moet rekenen. Ik hoop echter, dat de aanzet voor dit Basic-programma genoeg animo opwekt om ook zelf alvast eens aan de slag te gaan met de rest van het adresbestand. Zo niet, dan zul je tot de volgende Commodore-Info moeten wachten voor het tweede deel van dit programma.
Voor nu veel programmeerplezier en sterkte met de twee-dimensionale arrays. Het kan beslist geen kwaad om hier wat vingeroefeningen mee te doen.

Jan Bodzinga.

QSORT revisited

Dat het werken met Desktop Publishing je als redacteur niet altijd even gemakkelijk af gaat, bewees de Basic cursus van de vorige keer. Bij alle listings in die aflevering zijn helaas de tekens '<' en '>' weggevallen. Voor de beide listings van de Bubble-sorteermethode kan worden volstaan met het tussenvoegen van een '<' in regel 230. Dit teken moet staan tussen de beide variabelen A$(J) en A$(J+1). Ook een '>' kan hier worden toegepast, daardoor worden de gegevens in aflopende volgorde gerangschikt.
De Quicksort-listing geeft veel meer problemen. Daarom publiceren we de complete listing nogmaals.
J.B.

```
10    REM QUICKSORT
20    REM SNELLE BASIC-SORTEER ROUTINE
30    REM Jan bodzinga 1987
100   REM BEGIN PROGRAMMA
110   PRINT CHR$(147): REM CLS
115   PRINT: PRINT " QUICK-SORT DEMO"
120   INPUT "HOEVEEL STRINGS SORTEREN : ";N
130   X = RND(-TI) : REM SEED
140   DIM A$(N) : REM AANTAL
150   GOSUB 1000 : REM MAKEN STRINGS
199   REM INIT QUICK-SORT
200   PRINT "BEGIN SORTEREN"
210   DIM ST((LOG(N)/LOG(2)+4) , 1) :REM TEMP-ARRAYS
220   TI$="000000":REM TIJD
230   T = 1: ST(1,0) = 1 : ST(1,1) = N
240   L = ST(T,0) : R = ST(T,1): REM WAARDEN STACK
245   T = T - 1
250   X = L: Y = R: REM TELLERS
260   X$ = A$( ( L + R)/2):REM HELFT
300   IF A$(X) < X$ THEN X = X + 1 : GOTO 300
310   FOR V = 1 TO 10000
320   IF A$(Y) > X$ THEN Y = Y - 1 : NEXT V
330   V = 10000
340   IF X = Y THEN X = X + 1: Y = Y - 1 : GOTO 300
350   IF X < Y THEN GOSUB 1500 : GOTO 300 :REM WISSEL
360   IF X < R THEN T = T + 1 : ST(T,0)= X: ST(T,1) = R
370   R = Y
380   IF L < R THEN 250
390   IF T > 0 THEN 240
500   REM KLAAR MET SORT
510   TYD = TI/60
520   PRINT " KLAAR !!!"
530   FOR I = 1 TO N : PRINT A$(I) : NEXT
540   PRINT "TIJDSDUUR "; TYD;"SEC." 999 END
1000  REM GENEREREN STRINGS
1005  REM VIA RANDOM !!!
1010  I=0  : REM TELLER
1020  T$="": REM TEMP
1030  FOR I = 1 TO N
1040  FOR J = 1 TO 7 : REM AANTAL/STRING
1050  T$ = T$ + CHR$(65 + RND(1)*25)
1060  NEXT J
1070  A$(I) = T$: T$ = ""
1080  PRINT A$(I)
1090  NEXT I
1099  RETURN
1499  REM WISSEL
1500  TT$ = A$(X)
1510  A$(X) = A$(Y)
1520  A$(Y) = TT$
1530  X = X + 1
1540  Y = Y - 1
1550  RETURN
```

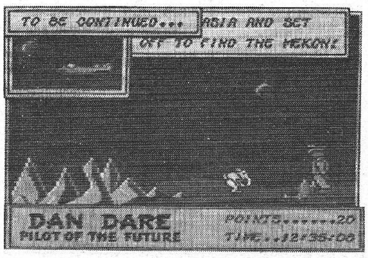

SOFT WIR WAR
DOOR BERT TIER.
AHUM!..
DE WINNAAR VAN ONZE WEDSTRIJD "BEDRIJFS BESPARING" IS GEWORDEN....
DE HEER GROOTKLOTE!!
APPLAUS!..
...DE HEER GROOTKLOTE WINT DE PRIJS VAN FL 5000,-!!...
..EN WEL MET EEN NIEUW ROVOLUTIONAIR BOEKHOUD PROGRAMMA!... WAT ONS BEDRIJF PER JAAR 2.4 MILJOEN BESPAART...
..EN!... 45 ARBEIDSPLAATSEN OVERBODIG MAAKT IN DE ADMINISTRATIEVE SECTOR VAN ONS BEDRIJF!!!..
...EN DAN GEEF IK U NU GRAAG DE GELEGENHEID DE HEER GROOTKLOTE TE FELICITEREN!!.....

64-Software

Dan Dare

Voor liefhebbers van de oude Buck Rogers comics is er nu een C-64 analogon: Dan Dare Pilot of the Future redt de aarde uit penibele situaties. Klink bekend in de oren? Dat klopt want dergelijke games gaan als shoot them up arcades per dertien in een dozijn. Dan Dare pretendeert echter iets meer te zijn.

Het verhaal is als een comic strip compleet met tekstbalken opgezet. Dan Dare paradeert tamelijk dandy-achtig in een fraai groen uniform over het scherm. Bovendien moet kolonel Dan aan het begin van het spel een gewichtige toespraak houden. Iets wat zo'n ijzervreter natuurlijk helemaal niet ligt. Gelukkig brengt een dringende oproep uitkomst. De aartsslechterik Mekom bedreigt de aarde met de meest vreselijke nucleaire projectielen en Dare's ingrijpen is dringend gewenst.



De semirealistische wereld van de Spidermannen, groene Hulken en en de Fantastische Vier wordt in dit Virgame aardig benadert. Toch vinden wij de papieren uitvoeringen wat leuker. Voor wie zich zelf eens met de joystick als comic-held wil uitleven vormt dit wellicht geen bezwaar.

Het game ontwikkelt zich als een grafisch adventure. Dan Dare, Prof Peabody, Dan's vriend Digby en het huisdier Stripey gaan met een futuristisch ruimtevaartuig Anastasia de boosaardige Mekom op zijn thuisbasis, een asteroide, te lijf. Het ontvangst pakt warm uit. De Treenhandlangers van Mekom, een soort enge lampjes op poten, die volgens de handleiding van runderachtige (?) oorsprong zijn, weten Dan's bemanning al meteen gevangen te nemen. Slechts Stripey wist te ontkomen.

Als een echte held staat de speler er dus direkt al in zijn/haar uppie voor. En de tijd dringt, Mekon blaast de aarde op als men zich niet voor de gestelde tijdslimiet overgeeft. Niettemin trekt Dan Dare onverschrokken ten strijde waarbij alle actie door tekstvensters worden toegelicht.

Echt moeilijk is het spel niet. Alleen het uitschakelen van de Treens en het opsporen van de drie verborgen toegangen tot Mekon's onder-grondse rijk, het bevrijden van de reisgenoten en het vinden van het allesvernietigende laserwapen vergen enige handigheid. Let wel even op de beschikbare energievoorraad tijdens de schermutselingen. Een uitgeputte Dan Dare komt immers niet ver! Lukt het Dan om zo'n Treen tijdig met een granaat te verschalken dan kost dit geen energie. Anders wordt het knokken op leven en dood.

Een aardig game voor liefhebbers van comic-strips. De echte avonturiers zullen zich echter vervelen. De prijs van deze C-64/ 128 strip bedraagt ongeveer *f* 35,-.

Leather Goddesses Of Phobos

Na de weekbladen, televisie en video moesten ook de computers aan de sexuele bevrijding geloven. Digitale sex schijnt, als wij de kranten berichten mogen geloven, behoorlijk populair en lucratief te zijn.

Sex Bulletin-Boards zorgden voor menig schandaal in Engeland en Frankrijk. Er kwamen zelfs rechtszaken (mishandeling en vrijheidsberoving, bemiddeling bij prostitutie) van. Overigens maken ook databanken en voorlichtingspakketten in de niet criminele sfeer opgang in de VS en Engeland. Een soort NVSH of Rutgersstichting op viditel dus. Een geheel andere ontwikkeling is de introductie van sex-videogames voor de Commodore. Strippoker, inmiddels ook in C-16 versie verkrijgbaar, en de Red Ligths of Amsterdam zijn daarvan enkele niet zo treffende voorbeelden. Het blijft allemaal wat tam. Inmiddels zijn echter ook Amerikaans softwarehuizen zich met de sex-markt gaan bezig houden. Zo vonden wij het Activision-pakket The Leather Goddesses of Phobos in ons (nog) maagdelijke postvak.

De leergodinnen van de marsmaan Phobos is een titel die veel belooft.

Het verhaal gaat ongeveer alsvolgt: De held zit in Upper Sandusky Ohio (een soort Lutjebroek in de VS) van een pilsje genieten. Op sexbeluste aliens staan echter voor niets en sleuren de nietsvermoedende (naar keuze) boerenkinkel of boerin met hun slijmerige tentakels in een ruimteschip. Hup naar Phobos waar een wellustige leergodin wacht. Deze leergodin wil eerst eens met deze testaardling experimenteren alvoren de gehele aarde te invaderen en tot een lustpark om te bouwen.

Natuurlijk weet de held (of heldin) na vele genoegelijke avonturen te onsnappen naar andere planeten. Ook daar heeft de Freudiaanse voortplantingsdrift echter toegeslagen en staan de bewoners de held naar eer en deugd. Gelukkig behoeft de held dit alles niet alleen te doorstaan want de trouwe vriend/vriendin Trent/Tiffany staan hem ter zijde.

Het uiteindelijke doel is een anti-leergodin machine uit verborgen componenten te construeren. Dat kan overigens lang duren als men zich onderweg te veel laat afleiden.

Het spel kent drie niveau's: tam, suggestief en ontuchtig. Daar kunt u de rest zelf wel bijdenken. Ter aanvulling van de, in vergelijking met tijdschriftfoto's niet al te sterke graphics levert Activision er nog een semi-3D comicboek bij.

Of een dergelijk pakket nu leuk is of niet kunnen wij niet beoordelen. Dat hangt van de gebruiker af. Activision zelf prijst de leergodinnen als vermakelijk aan.

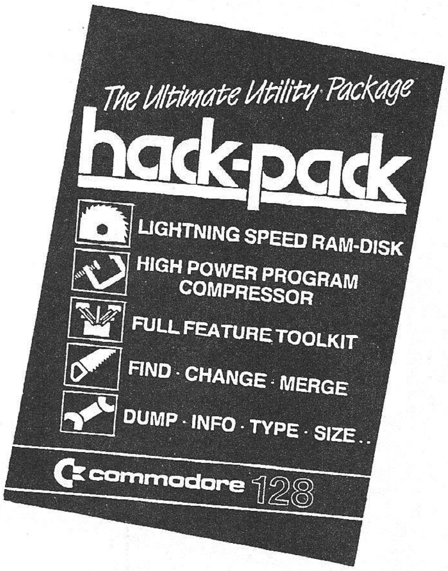
Ook de Duitsers trekken aardig van leer. Voor de liefhebber brengt het snuffelen in de kleine advertenties "toppers" als **Miss All Nude America** en **Girls They Want to have Fun** aan het licht. Sommige pakketten brachten het zelfs tot recenties in de mannenbladen. Voor zo'n 40 gulden kunt u bekijken of u deze sex-games ook leuk vindt. Ons inziens meestal weggegooid geld.

Hack-Pack C-128

Het Britse softwarehuis Systems Software brengt onder de naam Hack-Pack 128 een veelzijdige collectie utilities uit.

Het pakket biedt de gebruiker de volgende opties:

Een **RAM-disk.** Een deel van het C-128-geheugen wordt tot een virtuele diskdrive gemaakt. Deze drive (in het geheugen) is aanzienlijk sneller dan een standaard 1541 of 1571 drive,



maar heeft natuurlijk het nadeel van de toch beperkte geheugencapaciteit en het feit dat bij een stroomuitval alle gegevens verloren gaan. Een verstandige toepassing is het applicatie-programma in de RAM-disk te laden en de datafiles tijdig naar de conventionele drive weg te schrijven. De RAM-disk reageert op alle Commodore DOS-opdrachten zoals DLOAD, BLOAD, OPEN en CATALOG. Kopiëren verloopt via de COPY- opdracht. Middels enkele eenvoudige opdrachten kan de omvang van de RAM-disk gespecificeerd worden. Daarbij is het verstandig om geen grotere BASIC-programma's dan 32K in de RAM-disk te zetten om problemen met de allocatietabellen te voorkomen.

FIND, CHANGE en **MERGE** voor de bewerking van BASIC-strings en files.

DUMP, INFO, TYPE en **SIZE** voor het zichtbaar maken van variabelenlijsten, ASCII-files en het nog beschikbare geheugen.

Een **compressor** voor het beperken van de omvang, spaart geheugenruimte, van uw zelf gemaakte BASIC-programma's.

Het pakket doet in de praktijk wat het beloofd. Bijzonder lastig is de kopieerbeveiliging die het laden van de utilities bemoeilijkt en vertraagd. De prijs bedraagt circa *f* 110,-.

Trinity

Op 16 july 1945 verhief zich boven de "Journey of death", een stuk woestijn in New Mexico, de paddestoel van de eerste menselijke nucleaire explosie.

Een kleintje weliswaar, maar deze A-bom stond aan de wieg van de huidige bewapeningswedloop. De naam van dit experimentele atoombom-project was Trinity. In het gelijknamige tekstadventure legt Infocom de speler een intrigerende vraag voor. Zal hij of zij door in het verleden in te grijpen de nucleaire toekomst veranderen of niet?

Trinity is een epische odyssee op de grenzen van de realiteit. De speler voelt zich een soort Alice in Wonderland, maar dan heeft dat wonderland wel de beangstigende realiteit van de Twilight zone. Een alternatief universum openent zich. Magie en natuurkunde blijken in deze vreemde dimensie hand in hand te gaan. Elke nucleaire explosie blijkt de volgende te beinvloeden. Het is de taak van de held om regulerend in te grijpen alvorens de nucleaire holocaust de aarde verwoest.

Het begin van dit avontuur is rond-uit schokkend. Als nietsvermoedende toerist in de wereldstad Londen wordt de held(in) geconfronteerd met het loeien der sirenes en een allesvernietigende waterstofbom. Niemand kan aan dit Argameddon ontkomen of toch wel? Een mysterieuze stem roept en een witte deur opent zich naar een met paddestoelen begroeide magische wereld.

Infocom's sterontwerper Brian Moriarty op zijn best. Langs tal van intrigerende lokaties moet het raadsel ontsluierd worden. Ondanks de goede tekstparser is het adventure zelfs voor geroutineerde spelers moeilijk. Het plastische en sfeervolle Engelse taalgebruik maakt het er voor de Nederlander echt niet makkelijker op. Toch een aanrader voor wie eens iets geheel anders wil. Trinity is alleen verkrijgbaar in de C-128 versie. Het gebruik van de snelle 1571 diskdrive is aanbevolen. De prijs zal rond de 100 gulden komen te liggen.

●

Wanneer de computer een beetje kan praten, geeft dat vaak een hele nieuwe dimensie aan programma's. Ook voor de C-64 is spraaksynthese mogelijk, met speciale hardware-uitbreidingen of, wat minder van kwaliteit, met speciale software. Vaak is er een compbinatie van spreken en spraak herkennen, zodat de computer ook luistert, naar wat men zegt.

Sprekende computers



...MIJN MAN SPREEKT ALLEEN NOG IN MACHINETAAL MET MIJ..

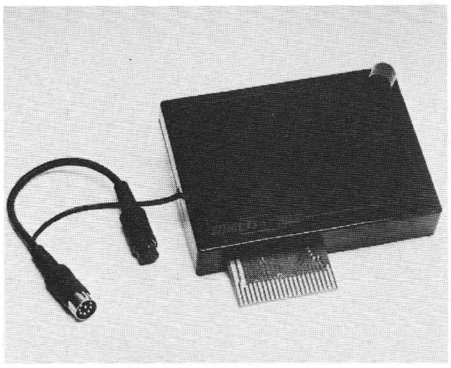
Spraak-interactie op de 64

De eerste keer, dat men in een auto stapt, die dan op vriendelijke toon zegt, dat de gordels niet vast zitten of de deur nog open staat, schrikt men even. Maar spraaksysnthese is langzamerhand zo gewoon geworden, dat men zich snel realiseert, wat er aan de hand is.
Deze techniek, die zijn oorsprong vond in het werk van de chipmakers, zoals Texas Instruments en General Instrument, heeft vele toepassingen. Iedereen kent wel de sprekende beertjes en ander speelgoed, maar ook op volwassen niveau komen we de spraakchips tegen. En voor gehandicapten is deze techniek natuurlijk ook een uitkomst.

Voor de Commodore zijn er al een paar jaar produkten op dit gebied, maar die worden gaandeweg wel steeds beter. In eerste instantie was het alleen mogelijk om via SID geluiden te maken. Dat klonk soms niet onaardig en is ook bij veel spelletjes gebruikt om het geheel wat levendiger te maken. Beter gaat dat, wanneer men een extra uitbreiding gebruikt. Door de juiste signalen op de connector te zetten, kan men dan de chips in de uitbreiding, eventueel met een microfoontje of aangesloten op de normale audio-uitgang tot spreken brengen, maar natuurlijk ook muziek laten maken. Voor het herkennen van spraak is altijd een uitbreiding nodig, de C-64 heeft uit zichzelf niet de mogelijkheid om audio-signalen te ontvangen.

Voice Master

Dit apparaat van Covox, dat in de VS ongeveer 280 gulden kost, bestaat uit een hardware-deel en natuurlijk ook uit speciale software om dat weer aan te sturen. Het totaal biedt niet alleen spraaksynthese, maar ook een beperkte vorm van spraakherkenning. Als extra is er dan ook nog een heel behoorlijk muzikaal talent aanwezig in de Voice Master. In tegenstelling tot de meeste andere produkten op dit gebied is het systeem zelflerend. Dat wil zeggen, dat men zelf woorden kan ingeven, die dan herhaald, maar ook herkend kunnen worden.



Zeker waar de meeste software op dit gebied uitgaat van de Engelse taal en Amerikaanse uitspraak, maakt dit het voor gebruik hier natuurlijk veel handiger. Om de hele zaak te programmeren biedt de Voice Master software een aantal nieuwe Basic commando's, die in gewone Basic programma's kunnen worden toegevoegd om electronische spraak en muziek te maken. Ook kan men wijsjes vastleggen in notenschrift door ze alleen maar te neuriën voor de microfoon van de Voice Master. Het programma genereert dan in beeld de daar bij behorende notenbalken, die men dan weer kan bewerken. Op die manier is het systeem heel handig voor wie zelf eens een wijsje wil componeren.

Firebird: Talking Teacher

Interactief leren met programma's, die ook echt tegen de "student" praten, is natuurlijk vooral bij jonge kinderen een goede aanpak. Firebird gebruikt daarbij de "Software Speech" technieken van Electronic Speech Systems en komt nu met een reeks educatieve programma's. Het pakket **Talking Teacher** is geschikt voor de 64/128, er is geen speciale hardware bij nodig, maar wel een diskdrive. Het doel van

het programma is om kinderen op een speelse manier kennis te laten maken met letters en woorden, waarbij de computer dan de juiste geluiden laat horen. Nog wel in het Engels, en dat betekent een iets andere uitspraak van bijvoorbeeld het alfabet dan bij ons. Prijs 40 pond.

Hearsay 1000

Een van de actiefste bedrijven op het gebied van spraaksynthese is HearSay. Hun hardware uitbreiding voor de Commodore 64/128 werkt dan ook met een brede range software-produkten van andere bedrijven. We noemen bijvoorbeeld Logo van Commodore, een aantal adventures van Infocom zoals Zork I,II, en III, Robots of Dwan van Epyx, Computer Diet van Scarborough, Face Maker van Spinnaker, en Consultant van Batteries Included.
Daarbij gaat het om de standaard software, men hoeft er geen speciale versie voor aan te schaffen. De computer spreekt de tekst-antwoorden van de computer nu uit, in plaats van ze alleen op het scherm te laten zien. Voor ongeveer 300 gulden kan men de Hearsay 1000 module kopen, in de laatste versie ook uitgerust met een "swift-load" optie om snelladen van software mogelijk te maken. Spreken, luisteren, gesproken commando's uitvoeren, met deze uitbreiding wordt de



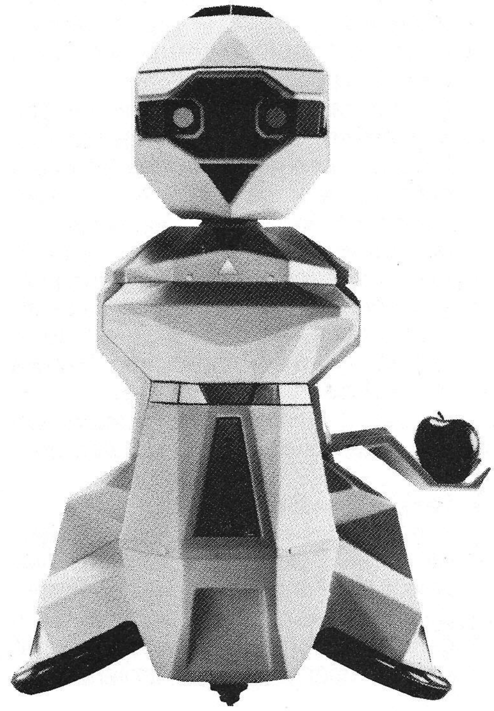
C-64/128 al een tamelijk brave huisslaaf, die doet wat hem gezegd wordt. Ook hier kan men met speciale Basic commando's allerlei effecten toevoegen aan eigen programma's en wie het helemaal goed wil doen, kan ook via Assembly opdrachten de Hearsay module laten spreken en luisteren.

Aqua Circus

Een van de software-pakketten, die speciaal door Hearsay zelf zijn gemaakt voor de Hearsay module is deze grappige introductie tot de computer voor kinderen van 3 tot 6 jaar. Kleurrijke plaatjes en bewegende animaties, alles met ondersteunende spraak (in het engels) leren de kleuter om basisvormen, letters en cijfers te herkennen. Ook eenvoudige sommetjes worden op een vriendelijke manier uitgelegd. ●

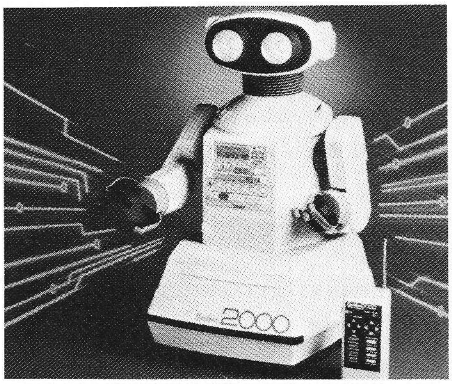
Informatie:
Hearsay, 1825 74th Street, Brooklyn, New York NY 11204 USA 718-2327266
Covox: 675D Donger Street, Eugene OR 97402 USA 503-3421271

In een van de vorige nummers van Commodore Info gaven wij u wat hardware-informatie over een niet alledaagse toepassing van de Userpoort, namelijk procesbesturing. Deze keer gaan we nader in op de software-kant van dit onderwerp. We beloofden u een besturingsprogramma om met behulp van de acht datakanalen een robot of iets dergelijks te kunnen besturen. We komen deze belofte nu na: RobotBasic.



De Userpoort op herhaling

Deel 2: RobotBasic



Bij het bepalen van de hardware componenten, stond het proces ook al voor een groot deel vast. We kozen voor zeven besturingsopdrachten in twee draairichtingen. U zult begrijpen, dat er een keus gedaan moest worden. Als dit niet uw keus was, dan kunt u met een beetje kennis van Basic 2.0 (die van de C-64 dus) zelf een configuratie samenstellen. Ook moet u natuurlijk een soldeerbout kunnen hanteren. Lezers die van nog uitgebreidere mogelijkheden gebruik willen maken, verwijzen wij naar de bekende electronicabladen. Hierin wordt ook zeer regelmatig aandacht besteed aan de computer. De invalshoek is natuurlijk wel anders in zo'n blad.

Wij willen het betrekkelijk eenvoudig houden, zodat iedereen (nou ja, bijna iedereen dan) mee kan doen. Ondanks die betrekkelijke eenvoud van ons projekt, is er toch veel mogelijk op procesbesturingsgebied met RobotBasic.

Het programma

Het programma heeft een aantal leuke bijzonderheden. Ten eerste is het helemaal geschreven in Basic. Dat betekent, dat het programma relatief langzaam is en de opdrachten in de bekende Basic 2.0 slakkegang worden uitgevoerd. Wij hebben daar natuurlijk het een en ander op bedacht en ook ingebouwd. Zo wordt de ingevoerde opdrachtenreeks voor de robot eerst in een verkorte code in het vrije RAM-geheugen (vanaf lokatie 49152) gezet. We hebben dat voor het gemak maar compileren genoemd. De uiteindelijke uitvoering van de opdrachten ondervinden daarna geen vertraging meer. De instructies worden rechtstreeks uit de geheugencellen geplukt. De positieve kant van Basic is, dat het toegankJijk is voor iedereen, die een beetje kan programmeren. Dat is met bijvoorbeeld machinetaal niet zo eenvoudig mogelijk.

Als u na een uurtje of wat ijverig intikken het programma start, dan lijkt het net, of de computer in de direkte mode staat. Het lijkt wel, of de computer een nieuw 'bedrijfssysteem' heeft! U kunt de cursor positioneren, het scherm wissen, de STOP-toets stopt alleen binnen het programma en niet Basic, e.d. Kortom, u kunt beschikken over alle functies van de scherm-editor en het toetsenbord. Dat komt, omdat er met een heel speciale INPUT wordt gewerkt. De string voor het INPUT-statement komt namelijk niet uit het toetsenbord, zoals gewoonlijk, maar vanaf het scherm. Daartoe moet er wel een file vanaf het scherm geopend worden om de gegevens op te halen. Omdat er toch nog wel eens iets fout wordt gedaan, zelf door de meest geroutineerde computeraar, heeft het 'bedrijfssysteem' zijn eigen nederlandse foutmeldingen. U kunt ze naar hartelust veranderen, uitbreiden, enz. Ook kunt u met een simpele 'HELP' opdracht even alle functies en mogelijkheden in juist formaat op het scherm zetten. Ook hierin valt veel naar eigen inzicht te veranderen. En dat is ook de bedoeling. De schil van het programma blijft goed draaien, terwijl u het programma aanpast aan uw eigen wensen.

Er zijn een aantal beperkingen. Het magische getal 255 (u weet wel, de C-64 is een 8-bitter!) mag nergens worden overschreden. Doet u dat toch, dan tikt het systeem u op de vingers. Dit is eigenlijk gedaan om het niet al te ingewikkeld te maken. Grotere getallen zouden dan weer om een omrekening naar low- en high-byte vragen en

dat wilden we voorkomen. Verder is een strakke afspraak ten aanzien van de syntax van de ingetikte opdrachten. Bij afwijken daarvan, meldt het systeem wederom zijn ongenoegen. U mag programmaregels gebruiken tussen 0 en 255. U kunt regels tussenvoegen en aanvullen, zoals u dat ook in Basic kan doen. Na het regelnummer moet u een spatie zetten. Daarna komt dan de eigenlijke opdracht.
Die opdracht moet bestaan uit een kanaalnummer (een nummer van een te besturen motor bijvoorbeeld), direkt gevolgd door een R of een L voor rechts of links draaien. Dan weer zonder tussenruimte de tijdsduur van de aktie. De ingegeven regel wordt direkt gecontroleerd op syntax-fouten en alleen geaccepteerd als alles in orde is. U kunt daarna meteen een nieuwe opdrachtregel invoeren, enz.
We zullen nu elk programmadeel aan een nader onderzoek onderwerpen.

Initialisatie

Om te beginnen de initialisatie. Eigenlijk spreekt dit deel voor zich. Hier worden de arrays voor de commando's en foutmeldingen opgezet. Er is gekozen voor arrays, omdat zo het zoeken naar en het controleren van het ingetoetste commando aanzienlijk wordt versneld. U zult dat zien in het bewuste deel van het programma.

Scherm-editor

De scherm-editor (vanaf regelnummer 500) zit ook betrekkelijk eenvoudig in elkaar. Er wordt een INPUT vanaf het scherm gehaald en geprint (en natuurlijk bewaard voor later gebruik!). Op deze manier blijven alle scherm-editor functies zoals scherm wissen en cursorbesturing behouden. U kunt zelfs programmaregels wijzigen met de cursortoetsen en dan op RETURN te drukken om de regel in te voèren. Dat gaat net zo als in Basic. Gemakkelijk in het gebruik dus.
Als de RETURN-toets is ingedrukt, gaat er een zoek- en controleroutine van start. Eerst wordt gekeken of het ingetoetste commando tot het repertoire behoort. Zo ja, dan wordt de betreffende aktie ondernomen. Is het een onbekend commando, dan wordt dat aan de gebruiker medegedeeld: ONBEKEND COMMANDO.

Opdracht analyse

Het is natuurlijk ook mogelijk, dat de invoer een programmaregel is. In dat geval springt het programma direkt naar een deel, waar de syntax wordt gecontroleerd. Die syntax ligt zoals gezegd vast, net als in elke andere programmeertaal. Vanaf regel 1000 wordt gekeken of de regel correct werd ingetypt. Om te beginnen moet de programmaregel beginnen met een regelnummer, net als in Basic. Dat regelnummer moet echter liggen tussen 0 en 255. Dan moet er een spatie staan. Bij weglaten van die spatie volgt weer een foutmelding. Dan volgt de analyse van de rest van de regel, de eigenlijke opdracht voor de Userpoort. Die opdracht moet beginnen met een getal tussen 1 en 7, overeenkomstig de zeven te besturen motoren van de robot. Een 8 op die plaats veroorzaakt een pauze van de aangegeven tijdsduur. Dan moet direkt daarachter staan of de motor linksom of rechtsom moet draaien. Dat wordt aangegeven met een R of een L. Andere letters worden geweigerd met een foutmelding. Meteen daarop aansluitend geeft u aan, hoe lang die motor in de aangegeven richting moet draaien. U gebruikt weer een getal tussen 1 en 255. Een programmaregel kan er zo uit zien:

10 1R20

In regel nummer 10 staat de opdracht voor motornummer 1 (aangesloten op datakanaal nummer 0 !) om 20 tijdseenheden rechtsom te draaien. De tijdseenheid, die wordt gebruikt is relatief en kan in de module voor de uitvoering van de opdrachtenreeks worden aangepast. De tijd staat in variabele U3.



Als alles correct is bevonden, wordt de betreffende regel definitief in de array geplaatst: in R1$(X) staat het regelnummer en in R2$(X) de opdracht.

Deze opslag is tijdelijk. In een later stadium moet u deze broncode compileren. De gecompileerde versie kan op diskette of cassette worden bewaard maar daarover later meer.

De broncode lijst

Als u de ingegeven commandoregels nog eens wilt bekijken en/of veranderen, kunt u gebruik maken van het commando BRON. Alle regelnummers verschijnen dan met daarachter de opdrachten. Dat gaat net zo als in het welbekende Basic, alleen veel langzamer. Alle elementen (van 0 tot 255) van de arrays moeten immers worden doorlopen! U kunt als u dat wilt ook een enkele regel laten zien door achter BRON het gewenste regelnummer te geven. Vergeet de spatie niet.

Het compileren van de broncode

Als de broncode (de opdrachtenreeks met de regelnummers) naar uw zin is, geeft u het commando COMP. Nu wordt de broncode omgezet in een reeks getallen, die vanaf lokatie 49152 in het geheugen worden gezet. Eerst worden er een drietal nullen in de eerste lokaties gezet om aan te geven, dat er een broncode is gecompileerd. Ook eindigt de reeks met drie toegevoegde nullen. Zonder broncode geeft dit commamdo weer een foutmelding, maar dat had u natuurlijk al verwacht.

De robot aan het werk

Het uur der waarheid is thans aangebroken. De aangesloten robot, in welke vorm dan ook, moet zijn werk gaan doen. Voorwaardelijk is een gecompileerde broncode in het geheugen. Zonder die getallen gaat het natuurlijk niet, met als gevolg een foutmelding. Als alles naar behoren is gedaan, worden de getallen voor de uitvoering uit de geheugencellen gehaald en geaktiveerd na het intoetsen van het commando DOEN. Het programmadeel zelf spreekt verder voor zich en begint op regel 5000. De functie van de toegevoegde nullen zal nu ook duidelijk worden.

De overige functies

Er blijven nog een aantal kleine stukjes programma over. Bij het geven van het commando HELP krijgt u alle commando's in beeld, samen met de betekenis of werking ervan. Dat gebeurt in het programmadeel dat begint op regel 8000.
Als er iets niet goed werd gedaan, wordt variabele F (van Fout) met een waarde geladen. Die waarde vormt de index van de foutmeldingen array F$(F) en produceert vanaf regelnummer 9000 aldus de foutmelding.
Als de broncode is gecompileerd, staan de instructies in de vorm van een getallenreeks vanaf 49152 in het geheugen, maar de broncode staat er ook nog! Dit betekent, dat u de instrukties kunt laten uitvoeren, evalueren en veranderingen aanbrengen in de broncode. Vervolgens kunt u weer COMP intikken en de broncode gaat opnieuw naar 49152 e.v. Voer de instructiereeks dan weer uit en kijk of alles goed gaat, net zo lang tot het naar uw zin is.
Een eenmaal goedgekeurde instructiereeks kan vanuit het geheugen naar diskette (of cassette) worden geschreven. Later kunt u die reeksen weer laden en uitvoeren. Helaas is er dan niets meer aan te veranderen, want de broncode is er niet meer. Ook nu moet u een naam aan de opdrachtenreeks geven om hem later herkenbaar te maken bij het inlezen. Geef die naam na het commando en vergeet de spatie niet. U krijgt voor deze handelingen de commando's SCHR voor schrijven (vanaf 11000) en LEES voor lezen (vanaf 12000). DISK geeft de directory van de diskette in drive 0 (device 8). Dit programmadeel loopt vanaf regelnummer 13000. Bij een eventule optredende diskettefout krijgt u alleen een foutmelding. Wat er fout ging, zult u zelf moeten uitzoeken. Daarin voorziet het 'bedrijfssysteem' niet. Een uitdaging misschien voor ambitieuze programmeurs.
Omdat de STOP-toets geheel niet meer werkt voor Basic, is ontsnappen naar Basic mogelijk gemaakt via het commando BASC. Wel wordt even gecontroleerd of u zich niet vergiste. Daarna is wel alles weg en u keert terug naar Basic door middel van een warme start.
Als u de broncode of een deel daarvan wilt verwijderen, dan kan dat met het VERW commando. Met een regelnummer erachter (weer een spatie!) wordt alleen de gegeven regel uit de broncode verwijderd. De start is op regel 7000.
Het programma werd geschreven voor diskettegebruik. Maar geen nood, cassettebezitters. Het geheel is zeer eenvoudig aan te passen. Het volgende moet daarvoor gebeuren. Verwijder regelnummers 11040 en 12020 en vervang ze door deze regels:

```
11040 OPEN 8,1,1,F$
12020 OPEN 8,1,0,F$:S1=49152
```

De rest kan dan gewoon blijven staan en het zaakje werkt nu met cassette. Wel krijgt u te maken met Engelse systeemverzoeken in dit geval, maar dat zal hopelijk geen problemen opleveren. De directory kunt u met cassette ook niet lezen. U kunt eenvoudigweg deze regel plaatsen en alle andere voor de directory-listing weglaten:

```
13000 RETURN
```

De directory opdracht blijft wel staan, maar wordt verder genegeerd. U kunt natuurlijk de gehele optie verwijderen, maar de voorgestelde werkwijze is eenvoudiger met duidelijk minder kans op fouten.
Dan nog even een extraatje voor de diskettegebruiker. Als de drive de file niet kan vinden (de drive was niet gereed, deurtje open, geen diskette, e.d.) zal de 1541-drive gaan klapperen en terugkeren naar de schermeditor met de melding: KLAAR...
Niks klaar, want de file of de directory werd niet ingelezen. U kunt in het betreffende programmadeel, direkt na het eerste GET-statement, deze regel invoegen:

```
IF ST=66 THEN F=8:GOSUB 9010:GOTO 12070
```

De systeemvariabele ST bevat namelijk de waarde 66 als er iets niet goed ging. Met gebruik van de inhoud van ST zijn trouwens nog veel meer beveiligingen aan te brengen. Wie durft?

Tenslotte

Nog even een paar opmerkingen tot slot. In regel 120 staat een REM-statement met daarachter een POKE-opdracht. Voer die regel eerst in met REM. Als alles goed loopt kunt u de REM verwijderen. De POKE-opdracht zorgt ervoor, dat de STOP-toets niet werkt in BASIC. Het programma zelf is dus niet meer te stoppen na die POKE-opdracht. Wel werkt de STOP-toets in RobotBasic normaal.

Er is tevens een mogelijkheid ingebouwd om de BRONcode listing tijdelijk te stoppen. Dat kan met de spatiebalk. Een tweede druk op die toets laat de broncode weer over het scherm rollen. Voor diskgebruikers is het wellicht handig om te weten, dat de zelfde werkwijze voor de directory listing geldt. Deze kan ook tijdelijk worden gestopt op deze manier, om later weer verder te gaan. De directory verstoort overigens de programma's in het geheugen niet.

Het gehele opdrachtenrepertoire nog even op een rij:

- BASC De gebruiker keert terug naar BASIC. Het programma gaat verloren.
- BRON Geeft de listing van de broncode. Alleen als die net is ingetypt.
- COMP Zet de broncode om in een getallenreeks (in RAM vanaf lokatie 49152).
- DISK Geeft de directory van de diskette in de drive.
- DOEN Voert de opdrachtenreeks uit. Die moet eerst zijn gecompileerd.
- HELP Geeft deze uitleg verkort op het scherm.
- LEES Leest een codereeks vanaf schijf of cassette.
- VERW Verwijdert de broncode of een regel daaruit.
- SCHR Zet een codereeks op diskette of cassette.

Een ieder, die zich geroepen voelt, dit programma zelf aan te passen of uit te breiden, wordt daartoe van harte uitgenodigd. Wij beseffen best, dat er nog vele verbeteringen en/of aanvullingen mogelijk zijn. Maar wij laten dat graag aan u over. U weet tenslotte zelf het beste, wat u wilt of nodig heeft. Wij hopen in elk geval, dat we u op een bruikbaar spoor hebben gezet. De gebroeders Marcel en Robert de Jonge uit Zierikzee zijn duidelijk aan het knutselen geslagen, getuige hun brief aan ons. Beste M en R, jullie denken wel in de juiste richting. Maar als je zo doorgaat, is/zijn jullie computer(s) wel binnenkort ter ziele. Een transistor werkt in theorie wel zo als jullie in het schema aangeven, maar in de praktijk zijn er nog wel wat meer componenten nodig. Gebruik het schema in geen geval in de praktijk! Er zijn wel enkele boeken over deze materie geschreven. Kijk maar eens in de betere boekhandel. Wij gaan in de toekomst door met het uitspitten van de Userpoort, zowel op hardware- als ook op softwaregebied, dus blijf lezen!

Voor alle knutselaars geldt natuurlijk: wees uitermate voorzichtig met aansluitingen en gebruik altijd inverters of buffers bij in- en uitgaande signalen. Wij kunnen natuurlijk geen verantwoordelijkheid dragen voor katastrofes, ook al verleiden wij u tot experimenteren. Als u onze aanwijzingen precies opvolgt, kan er echter niets misgaan en kunt u meer plezier aan uw computer beleven. En dat is uiteindelijk onze bedoeling. Tot zover, veel succes! ●

B.M.

Er zijn op de redactie een aantal reacties binnengekomen op onze oproep ten aanzien van een eventuele printservice. Het aantal is helaas te gering om werkelijk aan het aanmaken van printplaatjes van onze hardware-projecten te beginnen. Jammer voor alle knutselaars onder u. We blijven wel doorgaan met de hardware en zullen alleen werken met goed verkrijgbare onderdelen.

```
100 POKE 56579,255:POKE 56577,0
110 POKE 53281,14:POKE 53280,6
120 POKE 646,0:REM  POKE 808,234
130 PRINT CHR$(142)CHR$(8)
140 PRINT CHR$(147):PRINT
145 PRINT" **** C 64    ROBOT
    BESTURINGS CODE ****":PRINT
150 PRINT65535+FRE(0);
155 PRINT" BYTES BESCHIKBAAR":PRINT
160 PRINT" (C) 1986"SPC(5)"B.MUNNIKSMA"
170 OPEN1,0:DIM R1$(255),
    R2$(255),O$(11),C$(11)
180 :
190 FOR I=1 TO 8:READ F$(I):NEXT
200 FOR I=1 TO 9:READ O$(I)
210 READ C$(I):NEXT
220 :
230 DATA ONBEKEND COMMANDO
240 DATA FOUT IN SYNTAX
250 DATA SPATIE ONTBREEKT
260 DATA GESTOPT
270 DATA REGELNUMMER TE HOOG
280 DATA BRONCODE NIET GEVONDEN
290 DATA GEEN CODE IN GEHEUGEN
300 DATA DISKETTE FOUT
310 :
320 DATA BASC,TERUG NAAR BASIC
330 DATA BRON,TOONT BRON CODE
340 DATA COMP,ZET CODES IN HET GEHEUGEN
350 DATA DISK,GEEFT DISK BESTANDEN
360 DATA DOEN,VOERT HET PROGRAMMA UIT
370 DATA HELP,GEEFT DE FUNCTIES
380 DATA LEES,LADEN VAN DISK
390 DATA VERW,WIST HET GEHEUGEN
400 DATA SCHR,SCHRIJVEN NAAR DISK
410 :
420 REM SCHERM EDITOR
430 PRINT:PRINT:PRINT"KLAAR...":AN$=""
440 INPUT#1,AN$:PRINT
450 IF AN$="" THEN 440
460 IF VAL(AN$)<>0 THEN 1000
470 FOR I=1 TO 9
480 IF LEFT$(AN$,4)=O$(I) THEN 520
490 NEXT
500 IF AN$=O$(10) THEN F=6:GOSUB 9010
    :GOTO 430
510 F=1:GOSUB 9000:GOTO 430
520 ON I GOSUB 10000,2000,3000,13000,
    5000,8000,12000,7000,11000
530 GOTO 430
540 :
```

```
1000 REM OPDRACHT ANALYSE ***
1010 FOR I=1 TO LEN(AN$)
1020 IF MID$(AN$,I,1)=" " THEN 1040
1030 NEXT:F=5:GOSUB 9000:GOTO 430
1040 X=VAL(LEFT$(AN$,I))
1050 IF X>255 THEN F=5:GOSUB 9010:GOTO 430
1060 R1$(X)=LEFT$(AN$,I-1)
1070 R2$(X)=MID$(AN$,I+1)
1080 MN=VAL(R2$(X)):IF MN<1 OR MN>8 THEN
     F=2:GOSUB 9000:GOTO 430
1090 R$=MID$(R2$(X),2,1):IF R$="L" THEN
     R=128:GOTO 1120
1100 IF R$="R" THEN R=0:GOTO 1120
1110 F=2:GOSUB 9000:GOTO 430
1120 TT=VAL(MID$(R2$(X),3)):IF TT>255 OR
     TT<0 THEN F=2:GOSUB 9000: GOTO 430
1130 GOTO 440
1140 :
2000 REM BRONLIJST ROUTINE
2010 IF AN$="BRON" THEN 2050
2020 IF MID$(AN$,5,1)<>" " THEN F=2
     :GOSUB 9000:RETURN
2030 Y=VAL(MID$(AN$,6)):IF Y>255 THEN
     F=5:GOSUB 9000:RETURN
2040 PRINT R1$(Y)CHR$(32)R2$(Y):RETURN
2050 FOR I=1 TO 255:IF R1$(I)="" THEN 2100
2060 GET A$:IF A$="" THEN 2090
2070 IF ASC(A$)=3 THEN F=4
     :GOSUB 9010:RETURN
2080 IF ASC(A$)=32 THEN GOSUB 2120
2090 PRINT R1$(I)CHR$(32)R2$(I)
2100 NEXT:RETURN
2110 :
2120 REM WACHTEN
2130 GET A$:IF A$="" THEN 2130
2140 RETURN
2150 :
3000 REM COMPILATIE
3010 S=49152:FOR I=0 TO 2
3020 POKE S+I,0:NEXT
3030 S=S+3:FOR I=1 TO 255:LR=1
3040 IF R1$(I)="" THEN 3100
3050 IF MID$(R2$(I),2,1)="L" THEN LR=0
3060 POKE S,VAL(R2$(I))
3070 POKE S+1,LR
3080 POKE S+2,VAL(MID$(R2$(I),3))
3090 S=S+3
3100 NEXT:IF S=49155 THEN F=7
     :GOSUB 9010:RETURN
3110 FOR I=0 TO 2:POKE S+I,0:NEXT:S=S+3
3120 PRINT"COMPILATIE VOLTOOID":RETURN
3130 :
5000 REM UITVOERING PROGRAMMA
5010 S1=49152:IF S=0 THEN F=7
     :GOSUB 9010:RETURN
5020 S1=S1+3
5030 U1=PEEK(S1)
5040 U2=PEEK(S1+1)
5050 U3=PEEK(S1+2)
5060 IF U1+U2+U3=0 THEN POKE 56577,0:RETURN
5070 P1=2↑(U1-1):IF P1=128 THEN P1=0
     :P2=0:GOTO 5090
5080 P2=0:IF U2=1 THEN P2=128
5090 FOR T=1 TO U3:POKE 56577,P1+P2:NEXT
5100 S1=S1+3:GOTO 5030
5110 :
7000 REM VERWIJDEREN BRONCODE
7010 IF LEN(AN$)< 5 THEN 7030
7020 IF MID$(AN$,5,1)<>" "THEN F=3
     :GOSUB 9000:RETURN
7030 B=VAL(MID$(AN$,6))
7040 IF B=0 THEN RUN 170
7050 IF MID$(AN$,5,1)<>" "THEN F=3
     :GOSUB 9000:RETURN
7060 R1$(B)="":R2$(B)="":RETURN
7070 :
8000 REM HELPFUNCTIE
8010 FOR I=1 TO 10:PRINT CHR$(18)O$(I)
     TAB(8)CHR$(146)C$(I):NEXT
8020 RETURN
8030 :
9000 REM FOUTMELDING
9005 R1$(X)="":R2$(X)=""
9010 PRINT:PRINT CHR$(18)F$(F);
9020 IF F=8 THEN OPEN 15,8,15
     :PRINT#15,"I":CLOSE 15
9030 RETURN
9040 :
10000 REM TERUG NAAR BASIC
10010 PRINT:PRINT"ZEKER WETEN  J/N"
10020 GET A$:IF A$="" THEN 10020
10030 IF A$="J" THEN SYS 64738
10040 RETURN
10050 :
11000 REM SCHRIJVEN NAAR DISKETTE
11010 S1=49152
11020 F$=MID$(AN$,6):IF F$="" THEN F=8
      :GOSUB 9010:RETURN
11030 IF PEEK(S1)+PEEK(S1+1)+PEEK(S1+2)<>0
      THEN F=7:GOSUB 9010:RETURN
11040 OPEN 8,8,8,F$+",S,W"
11050 AD=(S-S1)/3:PRINT#8,STR$(AD)
11060 FOR I=S1 TO S-1
11070 A=PEEK(I):A$=STR$(A)
11080 PRINT#8,A$:NEXT
11090 CLOSE 8:RETURN
11100 :
12000 REM LEZEN VAN DISKETTE
12010 F$=MID$(AN$,6):IF F$="" THEN F=8
      :GOSUB 9010:RETURN
12020 OPEN 8,8,8,F$+",S,R":S1=49152
12030 INPUT#8,A$:AD=VAL(A$):S=S1+AD*3
12040 FOR I=S1 TO S-1
12050 INPUT#8,A$:A=VAL(A$)
12060 POKE I,A:NEXT
12070 CLOSE 8:RETURN
12080 :
13000 REM DIRECTORY
13010 OPEN 8,8,0,"$0"
13020 FOR I=0 TO 5:GET#8,A$
13040 NEXT
13050 GET#8,A$
13060 IF A$<>""THEN PRINT A$;:GOTO 13050
13070 PRINT:GET#8,A$
13080 GET S$:IF S$="" THEN 13110
13090 IF ASC(S$)=3 THEN 13190
13100 GOSUB 13200
13110 PRINT:GET#8,A$
13120 GET#8,L$:IF L$=""THEN L$=CHR$(0)
13130 GET#8,K$:IF K$=""THEN K$=CHR$(0)
13140 BL=ASC(K$)*256+ASC(L$)
13150 PRINT BL;SPC(5);
13160 GET#8,A$
13170 IF A$<>""THEN PRINT A$;:GOTO 13160
13180 GET#8,A$:IF A$<>""THEN 13080
13190 CLOSE 8:RETURN
13200 GET S$:IF S$="" THEN 13200
13210 RETURN
```

Vorige maand liet Bob Munniksma zien, dat het zelf maken van Basic-programmatuur ter ondersteuning van het leerproces in de klas geen onmogelijke opgave is. Deze keer volgen, zoals beloofd, de programmaregels, die het u mogelijk maken de resultaten van de leerlingen aan een nader onderzoek te onderwerpen.

Leerprogramma's maken in Basic

Deel 3: De administratie



Bij een goed leerprogramma behoort de mogelijkheid voor de leerkracht om een vinger aan de pols te houden. Bij een oefening op papier zijn er geen speciale voorzorgen nodig. Papier is geduldig en laat zich goed bewaren.

Indien een oefening op een computer gedaan wordt, komt er natuurlijk wel een en ander kijken als het gaat om correctie en analyse van de gedane oefening. Alle door de leerling ingetoetste antwoorden moeten ergens in de computer worden bewaard en wel zo, dat ze in een later stadium gemakkelijk boven water kunnen worden gehaald. Immers dan slechts kan de leerkracht de ontwikkeling van de leerling optimaal volgen. In de vorige Commodore Info kreeg u de bouwstenen voor een programma om spelling te oefenen. Deze keer breiden we dat programma uit met enkele administratieve modules. Alle resultaten blijven bewaard en kunnen worden bekeken en zelfs geprint.

Nodige aanpassingen

Aangezien het hele programma modulair is opgebouwd, is het erg eenvoudig een aantal modules toe te voegen. Een kwestie van uittypen en vastplakken via MERGE of APPEND als u beschikt over een goede toolkit. Heeft u die niet, dan kunt u het programma van de vorige maand laden en de nieuwe modules gewoon toevoegen. Niet vergeten vast te leggen als u klaar bent!

Weer zijn de modules gebaseerd op diskgebruik, maar evenals de vorige maand krijgen cassette-gebruikers de aanpassingen, zodat ze mee kunnen blijven doen. Over aanpassingen gesproken, het bestaande programma zal hier en daar een wijziging of aanvulling moeten ondergaan om de toegevoegde modules toegankelijk te maken vanuit de bestaande programmadelen. We zullen alle wijzigingen en aanvullingen precies aangeven, voorzien van commentaar. Ook komen de nieuwe modules natuurlijk uitgebreid aan de orde.

De administratie

Zoals gezegd, bij een toets- of oefenprogramma hoort een uitgebreide administratieve toepassing. Dan kan de leerkracht de resultaten beoordelen en aan een foutenanalyse onderwerpen om vervolgens de leerling, indien nodig, op het goede spoor te zetten met adequate oefenstof (wellicht ook per computer!). Ten eerste moeten alle ingetoetse antwoorden worden gecontroleerd en worden opgeslagen in het geheugen. Die controle zat al ingebakken in de oorspronkelijke versie van 'Raamwerk', de opslag niet definitief. Bij de volgende leerling was alles weer weg. Verder is het voor de leerkracht erg gemakkelijk, als de antwoorden op de ene of andere manier voorzien worden van een aanduiding 'goed' of 'fout'. De leerkracht kan dan in een oogopslag zien, waar de eventuele problemen zitten. Daarnaast zou het handig zijn om de gegevens ook nog op papier te kunnen zetten. U heeft dan wel een serieel aangestuurde printer nodig. Ook het definitief bewaren van de resultaten van de leerlingen, om eventueel later weer in te laden en te bekijken, te vergelijken of aan te vullen, is een handige mogelijkheid van de aanvulling die wij u nu op het leerprogramma uit het vorige nummer van Commodore Info geven. U merkt het wel, overal is aan gedacht.

Om te beginnen

De programmaregels van listing 1 moeten worden ingevoegd in het bestaande programma om de nieuwe modules toegankelijk te maken. Sommige regels zijn nieuw, andere zijn bestaande regels die een kleine wijziging ondergingen. Het aantal opdrachten in een oefening wordt teruggebracht tot 30. Dit houdt verband met de beeldschermpresentatie in de administratie. Er is gewoon geen ruimte op het scherm voor de weergave van 50 opdrachten.

In regel 52 stelt de variabele n het aantal plaatsen in de administratie voor. Dat aantal is overigens eenvoudig om te zetten voor meer of minder leerlingen. De programmadelen werken er even goed om. 75

LISTING 1

```
50 dim z$(30,2):print chr$(18);

52 x=1:n=75
53 dim n$(n),f$(n),z(n),g(n),w$(n,30)

137 print:print"3= Bestand laden"

145 on k goto 200,2000,9000

260 print left$(ne$,11)tab(3)"Aantal
    zinnen (max. 30) ";
270 if z>30 or z<1 then 200
```

```
LISTING 2

2005 gosub 3000:g=0:gs=0
2057 if gs=1 then f$=f$+"/G":goto 2085
2077 if q=3 then gosub 5000:goto 2055

2500 rem antwoord controle
2510 if an$=z$(a,2) then c$=" GOED ! ":
     g=g+1:w=43:gosub 2800:return
2520 c$=" FOUT ! ":w=45:gosub
     2800:return
2530 :
2800 zn=zn+1:w$(x,zn)=chr$(w)+an$
2805 for j=1 to 5
2810 print left$(ne$,3)chr$(18)c$
2820 for. w=1 to 100:next w:print
     left$(ne$,3)c$
2830 for w=1 to 100:next w,j
2840 print left$(ne$,3)"          "
2850 return

4002 n$(x)=n$:f$(x)=f$:g(x)=g
4003 z(x)=z:zn=0:x=x+1
```

leerlingen leek ons wel genoeg. De variabele x telt het aantal in de administratie aanwezige namen plus bijbehorende gegevens. U zult dat later zien.

Als u bijvoorbeeld 's morgens met een groep leerlingen oefeningen deed, en u wilt die bijvoorbeeld in de middaguren voortzetten, dan zult u de computer tussen de middag waarschijnlijk niet aan laten staan. Bij het uitzetten gaan de gegevens echter verloren. Dat probleem wordt opgevangen door de module van listing 8. Daarin wordt een bestand gemaakt van die gegevens. Als u het programma laadt en opstart, heeft u de keuze uit twee mogelijkheden. Daar komt door middel van de regels 137 en 145 een derde optie bij, namelijk het laden van het genoemde leerlingenbestand. Maakt u zich geen zorgen. Als dat bestand er niet is en u kiest toch voor die optie, gebeuren er geen rampen. U krijgt een passende melding op het scherm en het programma start opnieuw.

De resterende regels hebben weer te maken met het maximum aantal opdrachten in de oefening en hoeven geen nader betoog.

De tweede stap

Bij listing 2 gaat het om een aanvulling en een verandering van bestaande programmaregels. Regel 2005 zet variabelen op nul, terwijl 2057 een afgebroken oefening registreert in het systeem. De derde regel (2077) opent de mogelijk om de administratie binnen te gaan. In de invoer-module vanaf regel 1000 werd bij het indrukken van de functietoets f2 de variabele q voorzien van de waarde 3. Daar was reeds in voorzien in het oorspronkelijke programma. Als u nu tijdens de oefening op die functie-toets drukt, verschijnt de adminstratie op het scherm, met daarin de leerlingen die al aan de beurt zijn geweest, tesamen met de behaalde resultaten. Na het verlaten van de administratie keert dan het programma terug naar die regel met de opdracht GOTO 2055, en de oefening gaat gewoon verder. De administratie kan dus beschouwd worden als een zijsprong, in de vorm van een sub-routine. Het programmadeel vanaf regel 2500 is eigenlijk een gewijzigde module. U zult de structuur wel herkennen. Verwijder nu de programmaregels 2500 tot en met 2550 en zet die van de listing er voor in de plaats. U bent dan zeker van, dat de door de leerling ingetoetste antwoorden voor nader onderzoek bewaard blijven. Ze worden bewaard in een tweedimensionale array **w$**, waarbij **x** het leerlingennummer voorstelt en de variabele **zn** het nummer van de gedane opdracht. Als u dan in de module van de einde van de oefening (regels 4000 e.v.) ook nog de regels 4002 en 4003 toevoegt, dan worden ook nog naam, resultaten en gedane oefening van de bewuste leerling blijvend vastgelegd in het geheugen. Dat gebeurt overigens zo, dat de gegevens een vaste relatie krijgen voor later gebruik. De variabele x is weer het nummer van de leerling in de array.

Let even op programmaregel 2800. Daar wordt aan het gegeven antwoord een CHR$(W) toegevoegd. Als het antwoord goed is heeft **w** de waarde 43. Bij een fout antwoord is w 45. Neem de ASCII-tabel erbij en zie dat CHR$(43) het plusteken (+) is en CHR$(45) het minus-teken (-). Zo worden de antwoorden 'gemerkt'.

Extra kaders

De aanvulling in de vorm van listing 3 heeft slechts een cosmetische waarde. Hij zet op een bepaalde plaats op het scherm een kadertje, waarin vervolgens mededelingen kunnen worden gedaan aan de gebruiker. U zult dit nog tegenkomen.

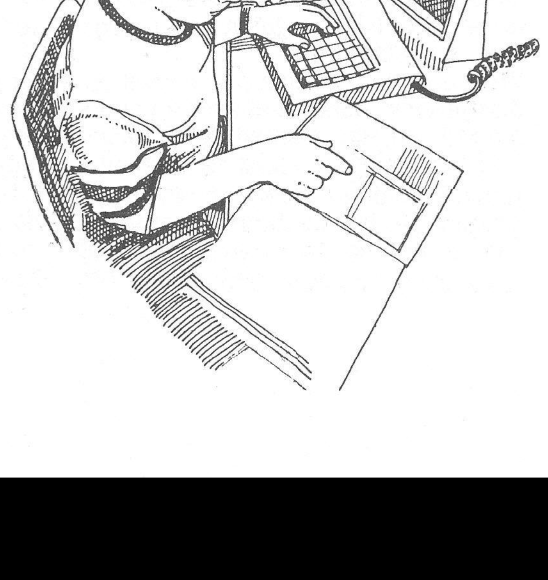
Tot zover de aanvullingen en voorbereidingen in het bestaande programma. We zullen nu overstappen naar de eigenlijke administratie.

Klasseoverzicht

Door tijdens de oefening op f2 (dat is SHIFT/ f1) te drukken, springt het programma naar de administratie. Deze begint op regel 5000. Zie daarvoor listing 4. Als de sprong wordt gemaakt, start de module met het tonen van de eerste 15 leerlingen, hun gemaakte oefening, het aantal opdrachten in de oefening en de score. Als een oefening om de een of andere reden is afgebroken, is de naam van de oefening in de lijst voorzien van de toevoeging ' /G '. Onderaan de pagina staan een aantal letters, die staan voor een even groot aantal functies. Met een druk op 'B' verschijnt de volgende pagina met weer 15 leerlingen, als die al geweest zijn natuurlijk. Het programma zal altijd stoppen bij de laatste leerling. Dus geen lege plaatsen. Zo kunt u 5 pagina's doorbladeren en ook weer terugbladeren als u op 'T' drukt. Een 'O' geeft de oefening terug, precies op de plaats, waar die werd verlaten. Die kan vervolgens gewoon worden voortgezet. Als u op 'D' drukt, worden alle gegevens, die tot dan zijn verzameld in een leerlingfile op diskette gezet. U kunt die in een later stadium dan weer inladen om de oefeningen voort te zetten.

U kunt de hele pagina op een printer afdrukken door op 'P' te drukken. Er volgt dan een sprong naar de module van listing 6. Daar wordt een schermdump gemaakt van de gegevens op de getoonde pagina.

De programmeertechniek van deze



module is slechts gebaseerd op het zichtbaar maken van de waarden in de gebruikte array's.
U kunt de hele pagina op een printer afdrukken door op 'P' te drukken. Er volgt dan een sprong naar de module van listing 6. Daar wordt een schermdump gemaakt van de gegevens op de getoonde pagina.
De programmeertechniek van deze module is slechts gebaseerd op het zichtbaar maken van de waarden in de gebruikte array's.

Schermdump

Vanaf regel 8000 worden de schermregels afgetast en op een printer afgedrukt. Staat de printer per ongeluk niet aan of off-line, dan meldt het programma dat en er loopt niets vast of zo. Bekijk de regels 8010 en 8020 om te zien hoe dat wordt gerealiseerd. Als alles goed was aangesloten wordt er een file naar het scherm geopend en de regels afgetast. De gevonden karakters op een schermregel worden in een string gezet en die string wordt naar de printer gezonden. Als de schermdump klaar is, komt u weer in de administratie terug. Alleen de regels, die feitelijke informatie bevatten worden gedrukt.

Leerlinggegevens

De laatste mogelijkheid in het klasseoverzicht is een druk op de letter 'L'. Als u dit doet, springt het programma naar de module van listing 5. Vanaf regel 6000 staat de mogelijkheid om de individuele prestaties van de leerlingen te evalueren, voor u open. Geef aan welk leerlingnummer u wilt zien en druk op return. Daarna kunt u overigens naar believen heen en weer bladeren door middel van de toetsen 'B' en 'T'. U krijgt steeds te zien: de naam van de leerling, de gedane oefening, het aantal opdrachten en het aantal goede antwoorden. Dat was al zichtbaar in het klasseoverzicht. Op de onderhelft van het scherm staan nu echter ook de gegeven antwoorden van de leerlingen. Ieder antwoord is voorzien van een merkje, om aan te geven of het antwoord goed of fout was. Ook nu kunt u de pagina in zijn geheel uitprinten, door op 'P' te drukken. U komt terug in de klasselijst door op 'K' te drukken. Vandaar kunt u de dan geldende opties weer kiezen.

LISTING 3

```
650 rem kaders
655 print left$(ne$,19);
660 print k1$;k2$;k2$;k2$;k2$;k3$;
665 print left$(ne$,21)tab(3);
670 return
```

LISTING 4

```
5000 rem administratie
5005 ll=0
5010 print chr$(147)chr$(18);
5015 print" Klasse-Overzicht  "
5020 print" Nr.   Naam"tab(20);
5025 print"Oef.    Aantal Goed   ";
5030 print mid$(k1$,2,38)
5035 if ll<15 then ll=0
5040 if ll>x then ll=0
5045 if ll<1 then ll=0
5050 for l=1 to 15:ll=ll+1
5055 if ll=x+1 then 5080
5060 y=1:if ll>9 then y=0
5065 print spc(y)ll;spc(2)n$(ll);
5070 print tab(20)f$(ll)tab(30)z(ll);
5075 print tab(35)g(ll):next l
5080 print left$(ne$,20)chr$(18)"B"
     chr$(146);
5085 print " = Bladeren"tab(20)chr$(18);
5090 print "T"chr$(146)" = Terugbladeren"
5095 print chr$(18)"L"chr$(146);
5100 print " = Leerlingen"tab(20)chr$(18);
5105 print "O"chr$(146)" = Oefening"
5110 print chr$(18)"D"chr$(146);
5115 print " = Disk"tab(20)chr$(18);
5120 print "P"chr$(146)" = Printen"
5125 gosub 800
5130 if a$="b" then 5010
5135 if a$="t" then ll=ll-30:goto 5010
5140 if a$="l" then gosub 6000:ll=ll-15:
     goto 5010
5145 if a$="o" then return
5150 if a$="p" then gosub 8000:ll=ll-15:
     goto 5010
5155 if a$="d" then gosub 9500:return
5160 if asc(a$)=13 then gs=1:return
5165 goto 5125
5170 :
```

LISTING 5

```
6000 rem leerlingen
6005 gosub 650:print"Welk nummer: ";
6010 i=3:gosub 1000:ln=val(an$)
6015 if ln<1 or ln>x then ln=1
6020 print chr$(147)chr$(18);
6025 print" Leerling nummer : "ln
6030 print left$(ne$,3);
6035 print tab(3)"Naam:  "n$(ln)
6040 print tab(3)"Oef. : "f$(ln)
6045 print tab(3)"Aant.: "z(ln)
6050 print tab(3)"Goed : "g(ln)
6055 print mid$(k1$,2,38)
6060 rem
6065 for o=1 to z(ln) step 2
6070 print w$(ln,o)tab(20)w$(ln,o+1)
6075 next:print left$(ne$,20);
6080 print chr$(18)"P"chr$(146);
6085 print " = Printen"tab(20)chr$(18);
```

```
6090 print "K"chr$(146)" = Klasselijst"
6095 print chr$(18)"T"chr$(146);
6100 print" = Terugbladeren";
6105 print tab(20)chr$(18)"B"chr$(146);
6110 print" = Bladeren"
6115 gosub 800
6120 if a$="p" then gosub 8000:goto 6020
6125 if a$="t" then ln=ln-1:goto 6015
6130 if a$="k" then return
6135 if a$="b" then ln=ln+1:goto 6015
6140 goto 6115
6145 :
```

LISTING 6

```
8000 rem scherm afdruk
8010 open 1,4,0:close 1
8020 if st<>0then8040
8030 goto 8090
8040 gosub 650
8050 printleft$(ne$,21)tab(8);
8060 print"PRINTER STAAT NIET AAN !"
8070 printtab(2);:gosub 910
8080 return
8090 print chr$(19)chr$(18);
8100 print"  P R I N T E N      Even
     Geduld a.u.b. "
8110 open 1,3:open 2,4,7
8120 print chr$(19)chr$(17);
8130 for pr=0 to 17:b$=""
8140 for pk=0 to 39:a$=""
8150 get#1,a$:if a$=chr$(13)then a$=" "
8160 b$=b$+a$
8170 next pk
8180 print#2,b$:next pr
8190 close 2:close 1:return
```

LISTING 7

```
9000 rem disk opslag laden
9005 open 1,8,8,"ldata,s,r"
9010 input#1,a$:x=val(a$)
9015 if st<>0 then close 1:goto 9045
9020 for g=1 to x
9025 input#1,n$(g),f$(g),z(g),g(g)
9030 for gg=1 to z(g)
9035 input#1,w$(g,gg):next gg,g
9040 close 1:gosub 900:goto 2000
9045 gosub 650:printleft$(ne$,21)tab(8);
9050 print"Geen bestand gevonden !"
9055 print tab(3);:gosub 910:run
9060 :
```

LISTING 8

```
9500 rem diskopslag schrijven
9505 open 1,8,8,"@0:ldata,s,w"
9510 print#1,x
9515 for g=1 to x
9520 print#1,n$(g)
9525 print#1,f$(g)
9530 print#1,z(g)
9535 print#1,g(g)
9540 for gg=1 to z(g)
9545 print#1,w$(g,gg):next gg,g
9550 close 1:return
```

De leerlingfile

Als u tussentijds of na afloop van de les de gegevens op diskette wilt bewaren, dan kan dat via de 'D'-optie in het klasseoverzicht. Alle gegevens, zoals naam, oefening en ook de ingetypte antwoorden per leerling worden dan in de file 'ldata' opgeslagen in verschillende array's. Listing 8 toont u de programma-regels. Ze spreken voor zich.
De regels van listing 7 laden de leerlingfile aan het begin van het programma, als dat werd gekozen. Het niet vinden van de file resulteert in een foutmelding en het opnieuw starten van het programma.

Tenslotte

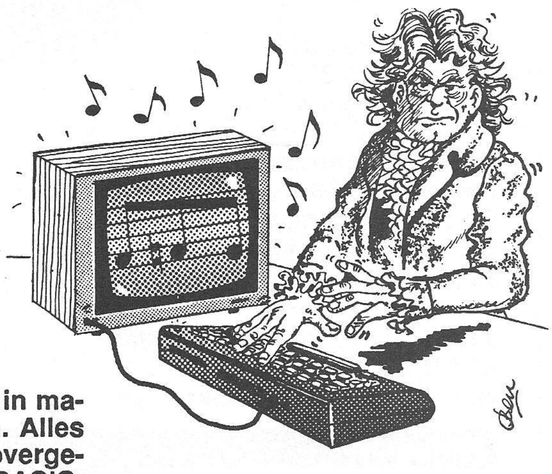
Bij het wegzetten van de leerlingfile keert u na het uitvoeren van de wegschrijf-handeling vanzelf terug naar de dan geladen oefening. Als u echter op <return> drukt, als u zich in de administratie bevindt, dan keert u terug naar de oefening met de waarde 1 in variabele gs. En zie wat er dan gebeurt na terugkeer in de leerling module (vanaf regel 2000):
Regel 2057 plaatst in de administratie voor de betreffende leerling een G achter de gedane oefening en breekt de oefening af. Dat is vooral erg handig als u bijvoorbeeld merkt, dat de leerling de oefening niet aankan. Het helemaal laten volgen van alle opdrachten is niet zinvol meer. De leerlinggegevens worden wel bewaard in de administratie! U ziet dan achter de gedane oefening de toevoeging /G. Deze 'ontsnapping' is echter alleen mogelijk vanuit het klasseoverzicht en niet vanuit de leerling-pagina's.
Als u met een cassette werkt, moet u de volgende regelnummer als volgt wijzigen:

9505 open 1,1,1,"ldata"
9005 open 1,1,0,"ldata"

De rest kan blijven staan. Of alles dan wel vlekkeloos zal verlopen kunnen wij natuurlijk niet garanderen. Wij zijn natuurlijk ook zeer benieuwd naar de ervaringen van cassettegebruikers. Een volgende keer zullen we eens bekijken, hoe we de engelstalige systeemmeldingen bij cassettegebruik, fouten en dergelijke kunnen onderscheppen om ze te vervangen door nederlandse informatie. Zeker als er kinderen met het apparaat werken, is dat onontbeerlijk.
Als laatste opmerking graag een aanwijzing voor wat minder ervaren programmeurs: als u geen ervaring heeft met in- en output statements, moet u even opletten bij de opdracht PRINT en INPUT. Aangezien PRINT# en PRINT door de computer geheel verschillend wordt opgevat, kunt u de laatste niet afkorten met een vraagteken gevolgd door een hekje. Het staat bij het listen wel correct op het scherm en toch ontmoet u bij de uitvoering van het programma een SYNTAX-ERROR melding. Het bekijken van de listing levert niets op, want u ziet niets bijzonders in de opdracht. Voer PRINT voluit in of gebruik de afkorting P SHIFT/R. Raadpleeg uw handleiding. Het zelfde verhaal geldt voor INPUT# en INPUT.
Als u een printer gebruikt, moet die wel via de seriële bus worden aangestuurd of via een interface op devicenummer 4 werken. Met diverse cartridges gaat het ook prima via de userpoort.

In onze serie over speciale effecten met de C-16 neemt R.Vogelaar ons deze keer mee op de muzikale toer. Met de TED chip is de C-16 namelijk zeer geschikt voor wat muzikale activiteiten.

Special effects op de C-16



Deel 3

Muziek maken met BASIC 3.5 is niet moeilijk. Als we het echter in machinetaal willen doen, komen we diverse moeilijkheden tegen. Alles moet nu met opdrachten, die rechtstreeks naar de geluids-chip overgebracht moeten worden. Deze worden gewoonlijk verricht door de BASIC-interpreter. Daarom moeten we iets weten van de geluidsregisters van de C-16 en over de manier, waarop Basic programma's in het geheugen staan.

Met deze machine kunnen we op 8 volume-niveaus en tweestemmig ongeveer 1020 toonhoogtes produceren, een voortreffelijke prestatie. Hier volgt eerst een korte herhaling met wat cijfers en adressen voor degenen, die een vorig artikel, enige nummers geleden, gemist hebben.

Het gaat om de adressen van de registers, die met geluid te maken hebben. dat zijn:

$FF0E = 65294 = Lo-byte stem 1
$FF0F = 65295 = Lo-byte stem 2
$FF10 = 65296 = Hi-byte stem 1 - verander alleen b1 en b0!
$FF11 = 65297 = Controlregister
b7,b3 geen functie
b6,b5,b4 resp.stem 3, 2, 1 aan als de bit "1" is
b2,b1,b0 volume 0...7
$FF12 = 65298 = Hi-byte stem 2 - verander alleen b1 en b0!
Stem 3 heeft dezelfde registers als stem 2.

Geluidsprogramma's

Er zijn een paar soorten geluidsprogramma's, te weten: de voorgeprogrammeerde en de vrij programmeerbare. Een programma van de eerste soort heeft een ingebouwd melodietje, bij de tweede soort kun je zelf opdrachten verstrekken. Op die tweede soort gaan we hier nu nader in.

Als je het programma wilt beïnvloeden, moet je informatie doorgeven. Daarom bekijken we eerst even hoe we dat kunnen doen, met behulp van een simpel BASIC-programmaatje. Tik het volgende maar eens in:

```
10 PRINT"C-16"
20 GOTO10
```

We gaan nu kijken hoe de te verwerken informatie (hier: de tekst "C-16") in het geheugen staat. Daarvoor hebben we het **MONITOR**-programma. Tik - vanuit de Monitor - in:

M 1000 101F (return).

Je krijgt nu een geheugendump van het begin van het BASIC-geheugen te zien. Die staat afgedrukt in bijgaand kader.

Kader 1 :
Beschrijving van de Bytes in het Basic geheugen van $1000-$3FF8

$1000		
1e byte =	$00	(altijd 00)
2e+3e byte =	$0D $10	($100D = beginadres volgende regel)
4e+5e byte =	$0A $00	($000A = dec. tien = regelnummer)
6e byte =	$99	(codegetal voor PRINT)
7e byte =	$22	(codegetal voor ")
8e-11e byte =	$43 $2D $31 $36	(tekst 'C-16')
12e byte=	$22	(codegetal voor ")
13e byte =	$00	(einde Basic regel)
$100D		
14e+15ebyte=	$15 $10	(beginadres volgende regel)
16e+17e byte =	$14 $00	($0014 = dec.20 = regelnummer)
18e byte =	$89	(codegetal voor GOTO)
19e+20ebyte =	$31 $30	(ASCII-waardes voor 1 en 0)
21e byte =	$00	(einde Basic regel)
$1015		
22e+23e byte =	$00 $00	(einde programma)

Als we nu RUN intypen, wordt elke byte uit het programma opgehaald door de CHARGET-routine, die in kader 2 staat.

Wie kunnen deze routine natuurlijk ook aanroepen om onze muziek-informatie uit het BASIC-programma te halen. Deze methode is dan ook toegepast in het grote machinetaal-programma, dat hierna volgt. De informatie kan de volgende inhoud en vorm hebben:

- Maatsoort:
  - MN = 2-kwartsmaat
  - MS = 3-kwartsmaat
  - ML = 4-kwartsmaat
- Toonlengte: L1...L8
- Pauzes: P1...P8
- Oktaaf: O1...O4
- Noten: ABCDEFG, al dan niet met een + voor de cis

Een voorbeeld

We zijn er nog niet, erst is nu een tamelijk fors stuk machinetaal nodig, dat u moet overtikken. In plaats van een disassembly is dat in de vorm van een geheugenuitdraai.

Na het intypen kan het geSAVEd worden met:

S"MUSIC", 01, 3A00, 3D80 (return) (eerst de Monitor aanroepen).

Later kan het geladen worden, vanuit de Monitor, met **L"MUSIC", 01** (return).

Als dat er in zit, kan het met relatief simpele middelen, zoals onderstaand Basic programma tot leven worden gebracht, de SYS(teemcall) zorgt daar voor. En dan kunt u met SYSsen beginnen!

```
5 VOL8
10 SYS 14848: REM START
   MACHINETAALPROGRAMMA
```

```
READY.

5 VOL8
10 V1=14965:V2=14977:E=15001:C=16381
20 SYS14848
100 PRINT"***********************************"
101 PRINT"*                                 *"
102 PRINT"*     RUIMTE                      *"
103 PRINT"*                   VOOR          *"
104 PRINT"*                                 *"
105 PRINT"*                         DE      *"
106 PRINT"*        MUZIEK                   *"
107 PRINT"*                                 *"
108 PRINT"*               OPDRACHTEN        *"
109 PRINT"*                                 *"
110 PRINT"***********************************"
999 END
1000 SYSE:IFPEEK(C)=0THENRETURN
1005 FORI=1TO6:NEXT
1010 GOTO1000

READY.
```

```
(EERST LADEN!)
100 SYS14965, "MS O3 L2
    ABCDEFG P3 GFEDCBA"
105 SYS14977, "MN O4 L1
    ABCDEFG P3 GFEDCBA"
110 GOSUB 1000
115 END
1000 SYS15001: IF PEEK(16381)=0
     THEN RETURN
1005FORI=1TO6:NEXT:GOTO1000
```

Versnellen van het stuk kan door in regel 1005 de 6 te verhogen. Als u stem 1 wilt starten, moet de SYS naar **14965** wijzen, voor stem 2 is dat **14977**. Het eigenlijke spelen gebeurt pas na **SYS 15001**, zie ook het voorbeeld.

Dan staat hier ook nog een listing, waarbij tussen de regels 100 en 900 een compositie is gestopt, in dit geval het Wilhelmus.

Veel succes met componeren!

R.Vogelaar

Kader 2

```
Monitor
PC   SR AC XR YR SP
;0000 00 00 00 00 F8

0473  E6 3B     INC $3B
0475  D0 02     BNE $0479
0477  E6 3C     INC $3C
0479  78        SEI
047A  8D 3F FF  STA $FF3F
047D  A0 00     LDY #$00
047F  B1 3B     LDA ($3B),Y
0481  8D 3E FF  STA $FF3E
0484  58        CLI
0485  C9 3A     CMP #$3A
0487  B0 0A     BCS $0493
0489  C9 20     CMP #$20
048B  F0 E6     BEQ $0473
048D  38        SEC
048E  E9 30     SBC #$30
0490  38        SEC
0491  E9 D0     SBC #$D0
0493  60        RTS
```

Het Wilhelmus

```
5 VOL8
10 V1=14965:V2=14977:E=15001:C=16381
20 SYS14848
100 SYSV1,"MS O2 L4 DGG L2 AB O3 C O2 A L4 B L2 AB O3 L4 C O2 B
    L2 AG"
101 SYSV2,"MS O3 L4 DGG L2 AB O4 C O3 A L4 B L2 AB O4 L4 C O3 B
    L2 AG"
103 GOSUB1000
105 SYSV2,"L4 A L8 G"
106 GOSUB1000
107 SYSV1,"O3 L4 DGG L2 AB O3 C O3 A L4 B L2 AB O3 L4 C O2 B L2
    AG L4 A L8 G"
108 GOSUB1000
109 SYSV1,"O2 L2 B O3 C L8 D L4 E L8 D L4 C O2 B L2 A"
110 GOSUB1000
111 SYSV1,"B L4 O3 C O2 BAG L8 A"
112 GOSUB1000
114 SYSV1,"L4 O1 D L2 G FGA L4 B L8 A L4 G FD L2 EFGG L4 GF L8 G"
115 SYSV2,"L4 O2 D L2 G FGA L4 B L8 A L4 G FD L2 EFGG L4 GF L8 G"
120 GOSUB1000
999 END
1000 SYSE:IFPEEK(C)=0THENRETURN
1005 FORI=1TO6:NEXT
1010 GOTO1000

READY.
```

Het Machinetaal-programma

```
MONITOR

  PC  SR AC XR YR SP
; 0005 00 10 00 02 F8

>3A00 A0 39 84 38 A0 3F 84 E1 :[illegible]
>3A08 A0 00 84 E0 98 91 E0 C8 :[illegible]
```

```
>3A10 D0 FB A9 0E 8D 00 3F A9
>3A18 0F 8D 40 3F EA A9 10 8D
>3A20 08 3F A9 20 8D 48 3F 60
>3A28 EA EA EA EA EA EA EA EA
>3A30 84 FF 20 73 04 08 A4 FF
>3A38 28 60 EA EA EA EA EA EA
>3A40 EA EA EA EA EA EA EA EA
>3A48 EA EA EA EA EA EA EA EA
>3A50 EA EA EA EA EA EA 20 30
>3A58 3A F0 19 C9 22 D0 F7 A5
>3A60 3B A0 01 91 E0 C8 A5 3C
>3A68 91 E0 20 30 3A D0 FB C8
>3A70 A9 01 91 E0 60 A9 3F 85
>3A78 E1 A9 00 85 E0 20 56 3A
>3A80 60 A9 3F 85 E1 A9 40 85
>3A88 E0 20 56 3A 60 EA EA EA
>3A90 EA EA EA EA EA EA EA EA
>3A98 EA A9 3F 85 E1 A9 00 85
>3AA0 E0 20 C1 3A A9 40 85 E0
>3AA8 20 C1 3A EA EA EA EA EA
>3AB0 EA EA A9 00 0D 03 3F 0D
>3AB8 43 3F 8D FD 3F 60 EA EA
>3AC0 EA A0 03 B1 E0 D0 01 60
>3AC8 AA CA D0 16 8A 91 E0 20
>3AD0 76 3C A0 04 B1 E0 D0 04
>3AD8 20 0C 3B 60 AA A9 00 91
>3AE0 E0 88 8A 91 E0 60 E6 B1
>3AE8 D0 02 E6 B2 60 A0 00 B1
>3AF0 B1 D0 01 60 C9 3A F0 FB
>3AF8 C9 20 D0 F7 20 E6 3A 4C
>3B00 ED 3A 20 ED 3A D0 01 60
>3B08 20 E6 3A 60 A0 01 B1 E0
>3B10 85 B1 C8 B1 E0 85 B2 20
>3B18 02 3B F0 2E EA C9 22 F0
>3B20 F6 C9 4C D0 03 4C 60 3B
>3B28 C9 4D D0 03 4C 79 3B C9
>3B30 4F D0 03 4C 9F 3B C9 50
>3B38 D0 03 4C C0 3B 38 C9 48
>3B40 B0 14 38 E9 41 30 0F 20
>3B48 DF 3B A0 01 A5 B1 91 E0
>3B50 C8 A5 B2 91 E0 60 A9 21
>3B58 EA EA 20 E6 3A 4C 17 3B
>3B60 20 02 3B D0 01 60 B0 03
>3B68 4C 56 3B E9 30 90 E7 0A
>3B70 0A 0A A0 05 91 E0 4C 17
>3B78 3B 20 02 3B D0 01 60 C9
>3B80 4E D0 04 A9 02 D0 11 C9
>3B88 53 D0 04 A9 03 D0 09 C9
>3B90 4C F0 03 4C 56 3B A9 00
>3B98 A0 06 91 E0 4C 17 3B 20
>3BA0 ED 3A D0 01 60 38 E9 31
>3BA8 90 04 C9 05 90 03 4C 56
>3BB0 3B 20 E6 3A 0A 0A 0A 0A
>3BB8 0A A0 07 91 E0 4C 17 3B
>3BC0 20 02 3B D0 01 60 38 E9
>3BC8 30 90 04 C9 08 90 03 4C
>3BD0 56 3B 20 E6 3A 0A 0A 0A
>3BD8 A0 03 91 E0 4C 4A 3B 0A
>3BE0 AA 20 ED 3A D0 01 60 C9
>3BE8 2B D0 08 20 E6 3A 8A 18
>3BF0 69 10 AA A0 07 8A 18 71
>3BF8 E0 AA A0 00 B1 E0 A8 BD
>3C00 01 3D 99 00 FF EA EA BD
>3C08 00 3D C0 0F F0 03 C8 C8
>3C10 C8 48 B9 01 FF 29 FC 99
>3C18 01 FF 68 C9 00 D0 07 49
```

```
>3C20 FF 39 01 FF D0 03 19 01
>3C28 FF 99 01 FF 20 32 3C 4C
>3C30 7C 3C A0 05 B1 E0 48 C8
>3C38 B1 E0 F0 13 AA 68 48 4A
>3C40 CA D0 FC A0 04 91 E0 68
>3C48 38 F1 E0 88 91 E0 60 A0
>3C50 04 A9 00 91 E0 88 68 91
>3C58 E0 60 C9 00 D0 0D A0 08
>3C60 B1 E0 49 FF 2D 11 FF 8D
>3C68 11 FF 60 A0 08 B1 E0 0D
>3C70 11 FF 8D 11 FF 60 A9 00
>3C78 20 5A 3C 60 A9 FF 20 5A
>3C80 3C 60 00 00 00 00 00 00
>3C88 00 00 00 00 00 00 00 00
>3C90 00 00 00 00 00 00 00 00
>3C98 00 00 00 00 00 00 00 00
>3CA0 00 00 00 00 00 00 00 00
>3CA8 00 00 00 00 00 00 00 00
>3CB0 00 00 00 00 00 00 00 00
>3CB8 00 00 00 00 00 00 00 00
>3CC0 00 00 00 00 00 00 00 00
>3CC8 00 00 00 00 00 00 00 00
>3CD0 00 00 00 00 00 00 00 00
>3CD8 00 00 00 00 00 00 00 00
>3CE0 00 00 00 00 00 00 00 00
>3CE8 00 00 00 00 00 00 00 00
>3CF0 00 00 00 00 00 00 00 00
>3CF8 00 00 00 00 00 00 00 00
>3D00 00 07 00 76 00 A9 01 06
>3D08 01 59 01 7F 01 C5 00 00
>3D10 00 3F 00 90 00 D8 01 30
>3D18 01 6C 01 A2 01 E5 00 00
>3D20 02 04 02 3B 02 54 02 83
>3D28 02 AD 02 C0 02 E3 00 00
>3D30 02 20 02 48 02 6C 02 98
>3D38 02 B7 02 D2 02 F3 00 00
>3D40 03 02 03 1E 03 2A 03 42
>3D48 03 56 03 60 03 71 00 00
>3D50 03 10 03 24 03 36 03 4C
>3D58 03 5B 03 69 03 79 00 00
>3D60 03 81 03 8F 03 95 03 A1
>3D68 03 AB 03 B0 03 B9 00 00
>3D70 03 88 03 92 03 9B 03 A6
>3D78 03 AE 03 B5 03 BC 00 00
?
```

Een rubriek met toppers uit het verleden, die anders te weinig aandacht krijgen. Sommige van de besproken games zijn nog te koop, voor andere zult u moeten snuffelen. Rob Goudriaan maakte de keuze en schreef de recensies.

Oud van GOUDriaan

Lode Runner.

Loderunner is een spel uit de wat oudere doos met nog steeds een trouwe schare aanhangers. Het is is uitgebracht door het software huis Broderbund. Ooit een tophit, en in bijvoorbeeld Japan nog steeds een succes.

Het is een spel dat elke keer weer uit de kast wordt gehaald en altijd weer boeit, omdat het zowel strategisch, als qua handvaardigheid toch wel erg goed is opgezet. De variatie is groot genoeg, en de moeilijkheidsgraad gaat net snel genoeg omhoog om het leuk te houden.

Het is een spel dat de nodige zweetdruppels en menig uurtje tijd kost en wat je niet zo maar eventjes tussen door speelt. Het aantrekkelijke van dit spel is dat men het toch al niet geringe aantal niveau's (levels), zo'n 150 stuks, zelf eenvoudig kan uitbreiden. Het spel begint in het rijk **Bungeling**. De leiders hiervan hebben de bevolking hun goudvoorrad afgenomen. Jij wordt er op af gestuurd om deze goudstukken terug te halen. Het grote gevaar zijn hierbij de bewakers.

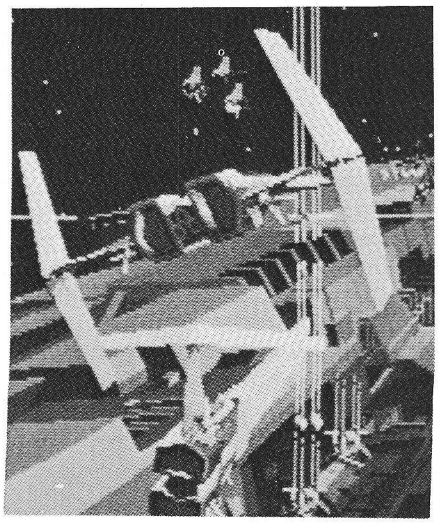
Men begint het spel op level 1 met vijf mannetjes (levens), bij het doorlopen van elk level krijgt men er een mannetje bij. De bedoeling is dat men de over de kamers verdeelde schatten verzameld. Denk hier niet te makkelijk over, want de vijandelijke troepen hebben overal hun spionnen neergezet.

Het enige wapen waar je over beschikt is een laser waar je gaten mee in de vloer kunt maken.

Hiermee kun je ook een bewaker uitschakelen. Elke gedode bewaker wordt echter weer door een nieuwe vervangen.

Als je alle schatten uit de kamer verzameld hebt verschijnt er bovenin een trap, waarmee je kan ontsnappen naar een volgende kamer. Dit geheel wordt begeleid door een vrolijk deuntje. Verschijnt deze trap niet dan kan je er zeker van zijn dat de bewakers nog een pakje hebben. Gelukkig is er geen tijdslimiet en zijn er ook speciale hulptoetsen. Met de run-stop toets wordt het spel tijdelijk stop gezet zodat je goed kan bekijken waar de schatten in de kamer staan en wat de beste route is. Ieder nieuw level start langzaam op via een diafragma effect.

Als je vindt dat alles te langzaam gaat is dit te versnellen met de + toets. Uiteraard kun je ook alles langzamer laten gaan door de - toets. Dit zijn de toetsen voor de (eerlijke) spelers die uiteindelijk in de top 10 terecht willen komen. Voor al de anderen of zij die wel eens verder willen kijken zijn er nog twee hulptoetsen n.l. de ctrl en F toets voor meerdere levens, maximaal 255 en de ctrl en de U toets waarbij je door gaat naar de volgende kamer.



Iedere kamer heeft verschillende verrassingen, zo heeft de ene kamer vloeren waar je zomaar door heen valt, of vloeren waar geen gat in te maken is. Moeilijk allemaal, kijk maar eens bij level 29, hangend aan een rekstok moet je gaten gooien zodat de zijkant opengaat snel het pakje pakken zodat je niet opgesloten raakt.

Een van de grootste plus punten van dit spel is dat men zelf ook levels kan maken. Loderunner was een van de eerste spelen met deze mogelijkheid en ook daarom een klassieker.

Revenge of the Mutant Camels.

Een onvervalste kraker is dit wat oudere spel, voor onze schietgrage klanten. Je moet niet gek staan te kijken als je ook sigaretten uit de lucht moet schieten die dan als as uit elkaar vallen.

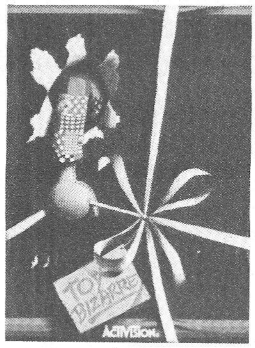
Voor wie niets te gek is heeft Jeff Minter dit spel gemaakt en ook deels zijn faam als programmeur gevestigd. Hij zegt zelf dat hij dit spel gemaakt heeft in nog geen twee maanden, na wel uren doorgebracht te hebben in de dierentuin om de gedragingen van de kamelen te bestuderen. Dit komt zeker tot uitdrukking in het spel. In de revenge ben je een kameel die allerlei vijandelijke golven moet doorstaan, niets is de vijand te gek. Vijanden zijn op allerlei mogelijke manieren vermomd. Het spel begint vrij onschuldig, grote volgels proberen je de doorgang te beletten, door gewichten naar beneden te gooien en je zo te verpletteren. Heb je dit overleeft, dan komen er de stortbuien met katten en honden naar beneden.

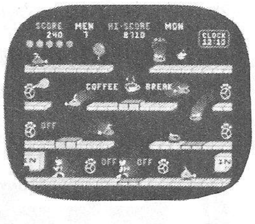
Wat verder op in het spel kom je springende en skieënde kangaroes tegen, lama's en telefooncellen die je proberen te verpletteren. Aggressieve theepotten proberen hun kokende inhoud over je heen te gooien.

Ook de computer wereld heeft model gestaan voor een aanvalsgolf, computers, joysticks en software viegen om je oren. En wat te denken van agressief klepperende wc potten en telefoons met klikkende kiesschijven.
Let ondanks al dit geweld ook eens op de achtergrond, dan zal je zien dat de bergen je af en toe een bemoedigende knipoog geven. Ook de muziek is niet vergeten, het spel begint met een geweldig stukje muziek.
Je begint het spel met vijf kamelen en elke kameel heeft een schild dat enkele aanvallen tegenhoudt, zodat je niet bij de eerste de beste aanval al het loodje legt.
Schieten kun je door middel van de joystick. Springen en bukken is ook mogelijk. Al met al zal het de nodige tijd vergen voor dat je alle gevaar zones hebt doorlopen.

Burgertime.

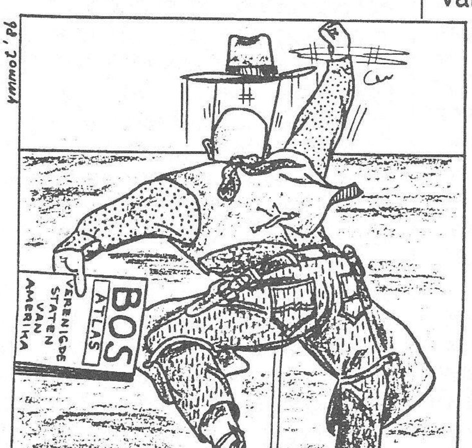
Heb je altijd al in een hamburger tent willen werken, maar daar nooit de kans voor gekregen, dan is dit een spel dat je deze kans geeft.
Per ronde moeten er vier hamburgers worden gemaakt. Voor de complete leken onder de hamburger koks, een hamburger bestaat uit vier onderdelen. Het onderste gedeelte is een broodje, dan een blaadje sla, dan de hamburger en tot slot het bovenste gedeelte van het broodje. Deze ingredienten liggen al klaar in de juiste volgorde, het enige dat je moet doen is er doorheen lopen.
De bedoeling is de hamburger onder op het scherm te krijgen. Het beste is om bovenaan te beginnen. Als er dan een onderdeel op een ander stuk valt, valt deze ook naar beneden. Maar zo eenvoudig is het natuurlijk niet, anders zouden er te veel hamburger makers rondlopen.
In onze hamburger zaak lopen er vreemd genoeg allerlei figuren rond die er normaal gesproken niet horen en die proberen jou van je taak af te houden. De spiegeleieren en de hotdogs kun je van je af houden door met peper te strooien, zodat ze tijdelijk even vast zitten. Natuurlijk is de peper voorraad daarbij beperkt. Maar alles is in deze zaak mogelijk, zo nu en dan verschijnt er op het beeld een milkshake en deze verschaft je dan weer een nieuwe voorraad peper. Het wachten in de zaak op de hamburgers vindt schijnbaar niemand erg, want je mag er net zolang over doen als je wilt. Extra punten levert het op als je een ei of hot-dog tussen je broodje krijgt. De graphics zijn erg eenvoudig maar toch wel aantrekkelijk.





Toys bizarre.

Dit is ook al een spel dat al enkele jaren geleden is uitgebracht door Activision. Nadat je dit spel bent begonnen wordt je ineens gepromoveerd tot de onderhoudsmonteur van een speelgoed fabriek.
Je naam is Merton. In de fabriek gaat er van alles fout en jij moet orde op zaken komen stellen. De produktielijn is op hol geslagen zodat het speelgoed een eigen leven gaat leiden. Na het laden van dit programma start je het spel met de F1 toets. Er komt dan een produktie platform op het beeld dat uit vier etages bestaat. Op elke etage staan ventielen die balonnen produceren. Als een ballon geheel is opgeblazen zweeft hij naar boven en knapt na drie seconden. Hierbij valt er een stuk speelgoed uit, dit speelgoed is bij het minste kontakt dodelijk en moet dus vernietigd worden. Het probleem is hoe vernietig je zo'n stuk speelgoed zonder dat je er zelf aangaat. Merton is niet een van de domste en ondekt dat als je er overheen springt, de dodelijke werking enige seconden wordt uitgeschakeld. Door hem op te pakken kan je hem dan vernietigen. Ook ontdekte hij dat je de ballon voor hij knapt kan aanraken, dan komt er geen speelgoed uit. Al met al moet je heel wat keren heen en weer lopen, maar je doet het voor de goede zaak, niet waar.
Om al dat geloop wat te beperken zijn er platforms gemaakt. Valt daar een stuk speelgoed op dan kan jij op een korresponderend platform springen, het blijft dan ongeveer 10 seconden verstijfd van schrik staan en is dus eenvoudig uit te schakelen. Dat ook hier weer niet alles zo makelijk is blijkt wel weer als je een rebelse dame, Hilda, tegenover je krijgt. Elke aanraking met haar is dodelijk dus geen dame om zonder handschoenen aan te pakken. Dat dit alles zwaar werk is had je al begrepen. Na een uur werken wordt je afgelost door je tweelingbroer Melvin. In werkelijkheid duurt 1 minuut op de klok twee seconden dus echt uitrusten is er niet bij.
Na het tweede veld zijn er bonuspunten te verdienen, er moeten acht ventielen dichtgedraaid worden, voor het uur werktijd over is. Hij wordt hierbij gehinderd door Hilda die ze weer opendraaid. Na het sluiten van het laatste ventiel wordt voor elke overgebleven minuut 50 punten toegekend. Ook is het mogelijk dat er een superbonus wordt uitgekeerd. Als U nog verder komt wordt het allemaal nog veel erger, Hilda roept er haar nicht bij, ook een Hilda. Ook is er een nep monteur de boel aan het saboteren, vast een neefje van Hilda, deze is niet bepaald vriendelijk. Nog lastiger wordt het als ook de vloeren op mysterieuse wijze onzichtbaar worden. Hierdoor wordt er een beroep gedaan op het geheugen van onze dappere monteur, waar kan ik wel en waar niet lopen. Bij elke 10.000 punten krijgt U er een leven bij. En als het U allemaal teveel wordt is door de run-stop toets het spel tijdelijk stop te leggen, om bijvoorbeeld echt even een kop koffie te drinken. Een spannend spel.



..USA! IK KOM ERAAN!!...

Print-out

Met o.a. Diskhulp - Torenspel - Checksum C-16 - Tempo typen C-16

In deze Print-Out rubriek hebben we een listing van Fons Reijsbergen opgenomen. Hij heeft daar een lange, maar uitsdtekende uitleg bij geschreven. We hebben deze uitleg volledig opgenomen. Iedere beginnende programmeur kan hier zijn voordeel mee doen. Het programma wordt regel voor regel uitgelegd. Hierdoor is het mogelijk zelf eenvoudig veranderingen in het programma aan te brengen.
De C-16 programma's in no.1 van dit jaar (het vorige nummer) hebben nogal wat problemen opgeleverd. We hebben het programma tempo typen 2 herhaald. U hoeft niet alles over te typen maar alleen de veranderingen erin aan te brengen.We houden ons altijd aanbevolen voor nieuwe programma's maar denk er dan aan altijd vermelden voor welke computer, niet te veel grafische tekens en niet te lang.
Voor vragen over de in Commodore Info verschenen listings blijft de listingtelefoon beschikbaar. Elke maandag zitten we van 17.00 tot 21.00 uur paraat voor vragen en opmerkingen op nummer 02155-25162. Denk er wel aan dat het niet mogelijk is om op andere tijden informatie over listings te krijgen, niet op bovenstaand nummer en ook niet bij de redactie.

Rob Goudriaan.

Inhoud van dit listingdeel

Syntax Checksum

Het overtikken van een listing kan een heel karwei zijn en als u een beetje normaal mens bent dan maakt u daarin beslist een aantal fouten. Nu is niets moeilijker om de fouten uit je eigen werk te halen. Al geruime tijd geleden heeft Jan Bodzinga hiervoor een zgn. **Checksum-programma** geschreven. Om de vele nieuwe lezers van Commodore-Info te helpen volgt hieronder nog een keer een volledige uitlegh over de werking van dit programma, waarmee het, hoe vreemd dat misschien ook lijkt, echt mogelijk is om met behulp van dit programma de fouten in elke door ons geplaatste listing op te sporen.
Hiervoor gaat u als volgt te werk:

❶ U tikt de listing heel zorgvuldig over en SAVEt hem voordat u het programma RUNt op een diskette of een cassette.

❷ U tikt het RUN commando in. Mocht het programma de boodschap **'Fout in dataregels!'** geven dan heeft u een fout bij het overtikken gemaakt. Herstel de fout en SAVE de verbeterde versie. Mocht het programma met de boodschap **'data is weggezet checksum testten met sys ....'** komen dan is tot dusver alles goed. Het programma is nu in een stukje machinetaalgeheugen gezet. Als u het NEW commando geeft blijft het toch in de computer staan.

Alle door ons geplaatste programma's zijn in Basic geschreven. Als u een programma heeft overgetikt SAVE het eerst, mocht er iets mis gaan dan hoeft niet de gehele listing opnieuw te gaan intikken. Als u nu een programma op fouten wilt gaan controleren dan kunt u dat in het geheugen laden (wel eerst het checksumprogramma hebben gerund). Vervolgens typt u zonder het programma te runnen de opdracht **sys 49152** (C-64) of **sys 1536** (C-16 en Plus/4) in.
Als alles goed is gegaan loopt er nu een rij regelnummers over het scherm met getallen erachter. Dezelfde lijst staat ook achter elk door ons geplaatste programma. Wijkt nu een nummer achter een regelnummer af van het nummer dat in het blad staat dan heeft u in die regel iets anders ingetikt dan er in het blad stond. U kunt de stroom getallen d.m.v. de RUN/STOP toets pauzeren en weer vervolgen met de F1 of F7 toets. Het is uitermate belangrijk dat u goed met dit programma overweg kunt en mocht U het niet goed werkend krijgen bel dan gerust even met onze listingservice telefoonlijn.

```
  1 rem ********************************
    **
  2 rem basic loader "SYNTAX.CHECKSUM"
  3 rem na de commando's 'run' en 'new'
  4 rem blijft dit programma in het ge-
  5 rem heugen. laad het te testen pro-
  6 rem gramma en tik daarna sys 49152.
  7 rem ********************************
    **
 10 i=49152  :rem beginadres
 20 reada:ifa<0then40:rem data ingeleze
    n
 30 pokei,a:i=i+1:b=b+a:goto20
 40 if b<>16844thenprint"[CLR-HOME]fout
     in dataregels!":b=0:end
 50 poke49184,148:poke49185,192
 55 i=49300
 60 read a: ifa<0then80
 70 pokei,a:b=a+b:i=i+1:goto60
 80 if b<>20068thenprint"[CLR-HOME]fout
     in dataregels! (vanaf regel 240)":
    b=0:end
 90 print"data is weggezet"
 95 print"checksum testen met sys49152"
100 data 165,43,166,44,133,163,134,164,
     169, 147
110 data 32,210,255,160,0,240,3,32,73,
     192
120 data 32,73,192,208,1,96,32,225,255,
      208
130 data 3,76,116,164,32,81,192,32,73,1
    92
140 data 240,12,201,32,240,247,24,101,1
    67,133
150 data 167,76,37,192,166,167,169,0,13
    2,168
160 data 32,205,189,169,13,32,210,255,1
    64, 168
170 data 76,17,192,200,208,2,230,164,17
    7,163
180 data 96,162,0,189,123,192,240,6,32,
    210
190 data 255,232,208,245,32,73,192,170,
    32,73
200 data 192,132,168,32,205,189,162,3,1
    69,32
210 data 32,210,255,202,208,250,169,0,1
    33,167
220 data 164,168,96,82,69,71,69,76,32,0
230 data -1
240 data 165,197,201,3,240,7,201,4,240
250 data 6,76,148,192,76,34,192,169
260 data 147,32,210,255,76,161,192
270 data -1

*** EINDE LISTING ***

    syntaxchecksum listtestprogramma

    regel 1    249
    regel 2    84
    regel 3    125
    regel 4    2
    regel 5    246
    regel 6    152
    regel 7    249
    regel 10    157
    regel 20    64
    regel 30    38
    regel 40    57
    regel 50    14
    regel 55    251
    regel 60    192
    regel 70    42
    regel 80    244
    regel 90    245
    regel 95    237
    regel 100    183
    regel 110    158
    regel 120    232
    regel 130    183
    regel 140    96
    regel 150    96
    regel 160    127
    regel 170    71
    regel 180    223
    regel 190    73
    regel 200    79
    regel 210    109
    regel 220    106
    regel 230    225
    regel 240    16
    regel 250    163
    regel 260    92
    regel 270    225

ready.
```

Diskhulp

Het volgende programma ontvingen wij van Marcel de Groot uit Hillegom en is een programma waar bij onze listingtelefoon veel om gevraagd wordt. Nadat het programma is geladen kunt u vanuit een menu alle lastige diskfuncties snel en foutloos uitvoeren. Maar ook het veranderen van het disk-id en de originele disk-header herstellen. Als u veel met uw diskdrive werkt kan dit pr0gramma u veel moeite besparen.

```
  1 rem disk-hulp / cbm-64 & 154
    1 drive
  2 rem door marcel de groot
  3 rem uit hillegom / 02520-166
    16
  4 rem
 10 dimn$(144),n(144):l$=chr$(0)
    :r$=chr$(13):poke53280,6:poke
    53281,6:p$="{2xspatie}"
 20 d$=r$+"druk op een toets.":s
    $="{20xspatie}":l=15:u1$="u1
     2 0 18 0"
 30 m$=chr$(8)+chr$(142)+"{CLR-HOME}
    {WIT}disk-hulp{13xspatie}doo
    r m.p. de groot":fori=1to40
 40 m$=m$+"#":next:printm$"{neer}
    maak uw keuze (x = stoppen):"
    :u2$="u2 2 0 18 0"
 50 print"{neer}a = directory"r$
    "b = validate een disk"r$"c
    = scratch ";:k$(1)="{RVS-aan}
    nee"
 60 print"een file"r$"d = verand
    er filenaam"r$"e = scratch m
    eerdere files"
 70 print"f = formatteren"r$"g =
     wis directory"r$"h = disk-n
    aam veranderen"
 80 print"i = disk-id veranderen"
    r$"j = originele disk-header
     herstellen"
 90 k$(0)="{RVS-aan}ja":poke649,
    1:poke198,0:wait198,1:geta$:
    a=asc(a$)-64:ifa<0thena=0
100 ifa=24thenprint"{CLR-HOME}":
    end
110 onagoto150,210,220,240,270,4
    20,460,480,520,560:open1,8,1
    5
120 print#1,"i"
130 input#1,e$,a$,b$,c$:print"{neer}
    disk-status: {RVS-aan}"e$","
    a$","b$","c$r$d$:close1
140 poke198,0:wait198,1:run
150 print"{CLR-HOME}directory":gosub
    700:open1,8,0,"$":get#1,a$,a
    $,a$,a$:gosub790
160 get#1,a$,a$:ifa$=""thenclose
    1:goto180
170 gosub790:c=c+b:wait197,64:goto
    160
180 print"{14xlinks} (="int(b/6.
    64+.5)"%).":ifc=664thenprint
    d$:goto140
190 print"{neer}werkelijke aanta
    l blocks free is"664+b-c"{links}
    !"
200 print"{neer}wilt u een valid
    ate (j/n)?":gosub620:open1,8
    ,15,"v":goto130
210 print"{CLR-HOME}validate":goto
    200
220 print"{CLR-HOME}scratch een
    file{neer}":print"filenaam: "
    ;:gosub640
230 print"{neer}wilt u {RVS-aan}"
    n$"{RVS-uit} scratchen (j/n)
    ?":gosub620:open1,8,15,"s:"+
    n$:goto130
240 print"{CLR-HOME}verander fil
    enaam"r$"{neer}oude naam{3xspati
    e}:";:gosub640:o$=n$:n$=""
250 print"{neer}nieuwe naam :";:
    gosub640:print"{neer}verande
    ren (j/n)?"
260 gosub620:open1,8,15,"r:"+n$+
    "="+o$:goto130
270 print"{CLR-HOME}scratch meer
    dere files"r$"{neer}director
    y wordt ingelezen.":gosub700

280 open1,8,0,"$":get#1,a$,a$,a$
    ,a$,a$,a$:a=0
290 get#1,a$:onlen(a$)goto290:get
    #1,a$,a$,a$,a$
300 get#1,a$:ifa$<>chr$(34)anda$
    <>""then300
310 ifa$=""thenclose1:a=a-1:goto
    340
320 get#1,a$:ifa$<>chr$(34)thenn
    $(a)=n$(a)+a$:goto320
330 a=a+1:goto290
340 print:fori=0toa:printn$(i)spc(
    20-pos(0))"scratchen (j/n)?{15xl
    inks}";
350 gosub730:k=1:print"{links}";
    :ifa$="j"thenk=0:n(i)=1:g=g+
    1
360 printk$(k)"{RVS-uit}{14xspatie}"
    :next:ifg=0thenprintd$:goto1
    40
370 print"{neer}validate na het
    scratchen (j/n)?":gosub730:if
    a$="j"thenv=1
380 print"{neer}doorgaan met scr
    atchen (j/n)?":gosub620:print
    "{neer}gescratcht zijn:"
390 open1,8,15:fori=0toa:ifn(i)then
    print#1,"s:"+n$(i):print"- "
    n$(i)
400 next:ifv=1thenprint#1,"v"
410 goto130
420 print"{CLR-HOME}formatteren{neer
    }"r$"disk-naam: ";:gosub640:
    o$=n$:n$="":print"{neer}id: "
    ;
430 l=1:gosub640:n$=","+n$:print
    "{neer}wilt u echt formatter
    en (j/n)?":gosub620
```

```
440 open1,8,15:print#1,"m-w"chr$
    (81)chr$(0)chr$(1)chr$(255):
    close1
450 open1,8,15,"n:"+o$+n$:goto13
    0
460 print"{CLR-HOME}wis director
    y{neer}"r$"disk-naam: ";:gosub
    640:print"{neer}wilt u de ";
470 print"directory echt wissen
    (j/n)?":gosub620:open1,8,15,
    "n:"+n$:goto130
480 print"{CLR-HOME}disk-naam ve
    randeren":gosub750:print"{neer}
    oude naam{3xspatie}: {RVS-aan}"
    left$(h$,16)
490 print"{neer}nieuwe naam : ";
    :gosub640:print"{neer}disk-n
    aam veranderen (j/n)?"
500 gosub620:open1,8,15:open2,8,
    2,"#":print#1,u1$:print#1,"b
    -p 2 144"
510 print#2,n$+left$(s$,16-len(n
    $));:print#1,u2$:close2:goto
    120
520 print"{CLR-HOME}disk-id vera
    nderen":gosub750:print"{neer}
    oude id{2xspatie}: "right$(h
    $,5)
530 print"{neer}nieuwe id:{9xspatie}
    (5 tekens){18xlinks}";:l=4:gosub
    640
540 print"{neer}id veranderen (j
    /n)?":gosub620:open1,8,15:open
    2,8,2,"#"
550 print#1,u1$:print#1,"b-p 2 1
    62":print#2,n$;:print#1,u2$:
    close2:goto120
560 print"{CLR-HOME}originele di
    sk-header herstellen"r$:open
    1,8,15:open2,8,2,"#"
570 print#1,u1$:print#1,"b-p 2 2
    00":get#2,a$
580 ifa$=""thenclose1:close2:print
    "header was al origineel{neer}"
    d$:goto140
590 h$=a$:fori=1to22:get#2,a$:h$
    =h$+a$:next:print"origineel:
     {RVS-aan}"h$
600 print"{neer}herstellen (j/n)
    ?":gosub620:print#1,"b-p 2 1
    44":print#2,h$;
610 print#1,"b-p 2 200":fori=1to
    23:print#2,l$;:next:print#1,
    u2$:close2:goto120
620 gosub730:ifa$="n"thenclose2:
    run
630 return
640 print"{RVS-aan}<{RVS-uit}{links}"
    ;:poke198,0:wait198,1:geta$:
    a=asc(a$+l$):b=len(n$)
650 ifl<>15andb<>l+1then670
660 ifa=13andb>0thenprint" ":return

670 ifa=20andb>0thenprint" {2xlinks}"
    ;:n$=left$(n$,b-1):goto640
680 ifb>lora=34or(aand127)<32then
    640
690 n$=n$+a$:printa$;:goto640
700 open1,8,15,"i":input#1,e$,a$
    ,b$,c$:close1
710 ife$<>"00"thenprint"{neer}di
    sk-status: {RVS-aan}"e$","a$
    ","b$","c$r$d$:goto140
720 return
730 geta$:ifa$<>"j"anda$<>"n"then
    730
740 return
750 gosub700:open1,8,15:open2,8,
    2,"#":print#1,u1$
760 print#1,"b-p 2 144":fori=1to
    23:get#2,a$:h$=h$+a$:next
770 print#1,"b-p 2 200":get#2,a$
    :ifa$<>""thenclose1:close2:return

780 print#1,"b-p 2 200":print#2,
    h$;:print#1,u2$:close1:close
    2:return
790 get#1,a$,b$:b=asc(a$+l$)+256
    *asc(b$+l$):printr$mid$(str$
    (b),2)p$;
800 get#1,a$:printa$;:onlen(a$)goto
    800:p$=" ":return

*** EINDE LISTING ***
```

REGEL 1	35	REGEL 300	137
REGEL 2	139	REGEL 310	2
REGEL 3	43	REGEL 320	91
REGEL 4	143	REGEL 330	109
REGEL 10	215	REGEL 340	253
REGEL 20	136	REGEL 350	22
REGEL 30	19	REGEL 360	36
REGEL 40	106	REGEL 370	69
REGEL 50	47	REGEL 380	245
REGEL 60	164	REGEL 390	177
REGEL 70	43	REGEL 400	182
REGEL 80	8	REGEL 410	29
REGEL 90	159	REGEL 420	106
REGEL 100	181	REGEL 430	88
REGEL 110	209	REGEL 440	219
REGEL 120	130	REGEL 450	78
REGEL 130	132	REGEL 460	175
REGEL 140	36	REGEL 470	176
REGEL 150	187	REGEL 480	8
REGEL 160	11	REGEL 490	140
REGEL 170	232	REGEL 500	91
REGEL 180	194	REGEL 510	221
REGEL 190	132	REGEL 520	165
REGEL 200	120	REGEL 530	228
REGEL 210	15	REGEL 540	123
REGEL 220	171	REGEL 550	181
REGEL 230	158	REGEL 560	219
REGEL 240	98	REGEL 570	86
REGEL 250	23	REGEL 580	17
REGEL 260	220	REGEL 590	47
REGEL 270	113	REGEL 600	110
REGEL 280	22	REGEL 610	127
REGEL 290	97	REGEL 620	210

```
REGEL 630   142        REGEL 730   109
REGEL 640   230        REGEL 740   142
REGEL 650   97         REGEL 750   93
REGEL 660   0          REGEL 760   3
REGEL 670   59         REGEL 770   155
REGEL 680   184        REGEL 780   76
REGEL 690   117        REGEL 790   46
REGEL 700   196        REGEL 800   202
REGEL 710   58
REGEL 720   142        READY.
```

Spyik Mini

Ingezonden door Gestel van Dimmen uit Kaatsheuvel.De bedoeling van dit spel is om een ruimteschip langs een aantal opstakels te sturen. Dit programma geeft duidelijk aan wat grafisch met een korte listing mogelijk is.

```
 0 poke53265,0
10 print"{BLAUW}{CLR-HOME}{RVS-aan}
   -{RVS-uit}'{RVS-aan}\I{RVS-uit}
   >{RVS-aan}P{RVS-uit}c{RVS-aan}
   N{RVS-uit}a{RVS-aan}PI{RVS-uit}
   ={RVS-aan}P{RVS-uit}c{RVS-aan}
   .{RVS-uit}a{RVS-aan}PI{RVS-uit}
   7{RVS-aan}P{RVS-uit}c{RVS-aan}
   .{RVS-uit}'{RVS-aan}PI{RVS-uit}
   ;{RVS-aan}P{RVS-uit}c{RVS-aan}
   N{RVS-uit}'{RVS-aan}P{RVS-aan}
   -E{RVS-uit}'{RVS-aan}I{RVS-uit}"
   ;
20 print"~{RVS-aan}P{RVS-uit}c{RVS-
   aan}N{RVS-uit}a{RVS-aan}PI{RVS-u
   it}'{RVS-aan}P{RVS-uit}c{RVS-aan
   }.{RVS-uit}a{RVS-aan}PI{RVS-uit}
   chr$(34)+chr$(34)"{links}{RVS-aa
   n}P{RVS-uit}c{RVS-aan}.{RVS-uit}
   '{RVS-aan}PI{RVS-uit}}{RVS-aan}
   P{RVS-uit}";
30 print"{RVS-uit}c{RVS-aan}N{RVS-u
   it}'{RVS-aan}P-{RVS-uit}{RVS-aa
   n}PI{RVS-uit}a{RVS-aan}P{RVS-uit
   }a@{RVS-aan}H(@?P;@{RVS-uit}
   d{RVS-aan}P6@{RVS-uit}'L'{RVS-aa
   n}p{RVS-uit}'":v=53248:s=542
   72
35 forl=0to24:pokes+l,0:nextl:poke
   s+24,15:pokes+5,0:pokes+6,24
   8:pokes+4,17
40 forl=0to87:poke36864+l,peek(
   1024+l):nextl:poke36933,96:poke
   36936,224
50 poke36944,160:sc$="00000":hi
   $="00000":poke53280,6:poke53
   281,0:poke649,1
60 data0,40,0,0,174,0,0.174,0,0
   ,174,0,5,85,80,94,149,173,21
   ,85,84,0,235
```

```
 70 forl=0to22:reada:poke12288+l
    ,a:nextl:forl=0to42:poke1231
    1+l,0:nextl
 80 pokev+21,0:pokev+1,60:pokev,
    35:pokev+28,255:pokev+29,255
    :poke2040,192
 90 pokev+39,10:pokev+38,1:pokev
    +37,2:poke36941,3:sc=0:poke5
    3265,0
100 a=31:print"{BLAUW}{CLR-HOME}"
    tab(a)"{BLAUW}{RVS-aan}{9xspati
    e}":printtab(a)"{RVS-aan}{op}
     {WIT}{RVS-uit}O77777P{BLAUW}
    {RVS-aan} ";
110 printtab(a)"{RVS-aan} {RVS-uit}
    {WIT}4score*{BLAUW}{RVS-aan}
     ":printtab(a)"{RVS-aan}{op}
     {WIT}{RVS-uit}4"sc$"'{BLAUW}
    {RVS-aan} ";
120 printtab(a)"{RVS-aan} {RVS-uit}
    {WIT}L/////:{BLAUW}{RVS-aan}
     ":printtab(a)"{RVS-aan}{op}
    {9xspatie}";
130 printtab(a)"{RVS-aan}{9xspatie}"
    :printtab(a)"{RVS-aan}{op} {WIT}
    {RVS-uit}O77777P{BLAUW}{RVS-aan}
     ";
140 printtab(a)"{RVS-aan} {RVS-uit}
    {WIT}4high *{BLAUW}{RVS-aan}
     ":printtab(a)"{RVS-aan}{op}
     {WIT}{RVS-uit}4"hi$"'{BLAUW}
    {RVS-aan} ";
150 printtab(a)"{RVS-aan} {RVS-uit}
    {WIT}L/////:{BLAUW}{RVS-aan}
     ":printtab(a)"{RVS-aan}{op}
    {9xspatie}";
160 forl=0to8:printspc(a)"{RVS-aan}
    {BLAUW}g{l.blauw}a{CYAAN}m{WIT}
    e o{CYAAN}v{l.blauw}e{BLAUW}
    r";:nextl:print"{op}"
170 printtab(a)"{op}{5xrechts}{WIT}
    >{BLAUW}{RVS-aan}{3xspatie}"
    :printtab(a)"{5xrechts}{WIT}
    {op}uit{RVS-aan}{BLAUW} "
180 printtab(a)"{op}{5xrechts}{WIT}
    >{BLAUW}{RVS-aan}{3xspatie}"
    ;:printtab(a)"{RVS-aan}{BLAUW}
    {8xspatie}{HOME}":poke2023,1
    60
190 poke56295,6:forl=0to30:poke1
    024+l,119+rnd(1)*2:nextl
200 forl=0to30:poke1984+l,247+rnd
    (1)*2:nextl
210 a=0:forl=0to22:poke1064+a,10
    1:poke55336+a,1:a=a+40:nextl

220 a=30:forl=0to19:poke1064+a,1
    03:poke55336+a,1:a=a+40:next
    l
230 print"{HOME}{3xneer}{8xrechts}
    {l.rood}dimmen presents"
240 printtab(10)"{2xneer}{ROOD}s
    pyik mini."
250 printtab(8)"{2xneer}{GEEL}jo
    ystick port 2":printtab(10)"
    {neer}{GROEN}or keyboard"
```

```
260 print"{l.groen}{neer}{rechts}
    up:{CYAAN}u{2xspatie}{l.groen}
    down:{CYAAN}n{2xspatie}{l.groen}
    left:{CYAAN}h{2xspatie}{l.groen}
    right:{CYAAN}j"
270 printtab(9)"{2xneer}{PURPER}
    try to escape.":poke53265,27

280 pokes+1,rnd(1)*23:geta$:ifpeek
    (56320)=127anda$=""then280
285 le=-1:gosub360
290 pokev+21,255:pokev+31,0:poke
    s+4,87:pokes+1,5
300 print"{HOME}{10xneer}":a=31:
    forl=0to8:printspc(a)"{RVS-aan}
    {ROOD} {l.rood} {GEEL} {WIT}
    {3xspatie}{GEEL} {l.rood} {ROOD}
     ";:nextl
320 sys9*4096:ifpeek(v)<10thengoto
    360
330 pokes+4,129:forl=0to50:a=rnd
    (1)*16:pokes+1,a:poke53281,a
    :nextl
340 pokes+4,17:poke53281,0:ifval
    (sc$)>val(hi$)thenhi$=sc$
350 geta$:goto80
360 print"{HOME}{neer}{RVS-uit}{rech
    ts}{28xspatie}"
361 print"{HOME}{2xneer}{RVS-uit}"
    ;:forl=0to10:print"{rechts}{29xs
    patie}";
365 printspc(11)"{29xspatie}":next
    l:pokev+1,60:pokev,35
370 le=le+1:onlegosub410,460,480
    ,500,530,410,460,480,500,530
    ,540,630
380 ifle=0thengosub480
390 sc=sc+35:sc$=right$("00000"+
    right$(str$(sc),len(str$(sc)
    )-1),5)
400 print"{HOME}{3xneer}{WIT}{RVS-ui
    t}"spc(33)sc$:goto290
410 print"{HOME}{BLAUW}{4xneer}{rech
    ts}{GEEL} jean michel jarre{2xsp
    atie}:":print"{3xneer}"spc(2
    0)"{oranje}music 1986"
420 gosub450:printspc(15)"{4xneer}
    {GEEL}rendez vous !!!":gosub
    450
440 forl=0to50:pokes+1,l:nextl:return

450 print"{7xrechts}{WIT}.{8xrechts}
    .{4xlinks}{neer}.{3xrechts}.
    {neer}{5xrechts}.{neer}{5xlinks}
    .":return
460 print"{HOME}{neer}{6xrechts}
    {6xspatie}{l.groen}queen{4xspati
    e}":gosub450:gosub450:gosub4
    50:gosub450:gosub450
470 print"{neer}{5xrechts}{GROEN}
    it's a kind of magic.":return

480 print"{HOME}{2xneer}{8xrechts}
    {BLAUW}c= commodore info":print
    "{neer}{10xrechts}{RVS-aan}{GROE
    N}{2xspatie}{l.groen}{2xspatie}
    {l.blauw}{2xspatie}{l.rood}{2xsp
    atie}{CYAAN}{2xspatie}{GEEL}
    {2xspatie}"
490 forl=0to8:printtab(12)"{ROOD}
    U{l.rood}@@@@@{ROOD}I":nextl
    :return
500 print"{HOME}{8xrechts}{2xneer}
    M{neer}M{neer}M{neer}M{neer}
    M{neer}M{ROOD}N{op}N{op}N{op}
    N{op}N{6xneer}"
510 printtab(12)"{BLAUW}]{neer}{link
    s}{neer}]{neer}{links}{neer}
    ]{neer}{links}{neer}]{neer}{link
    s}{neer}]{neer}{links}{neer}
    ]{6xop}{4xrechts}{WIT}.{2xneer}
    .{2xneer}.{2xneer}.{2xneer}.
    {op}.{2xop}.{2xop}.{2xop}.{2xop}
    ."
520 return
530 print"{HOME}{neer}{2xrechts}
    {8xspatie}{bruin}bonus stage
    .{8xspatie}":return
540 print"{HOME}{neer}{2xrechts}
    {9xspatie}{grijs 3}machines.
    {8xspatie}":poke36941,2
550 print"{l.rood}{neer}{9xrechts}
    it's software":print"{neer}{9xre
    chts}{grijs 3}it's hardware"

560 print"{7xrechts}{WIT}.{neer}
    {3xrechts}.{neer}{3xrechts}.
    {neer}{4xrechts}.{neer}{5xrechts
    }.":print"{neer}{l.groen}{10xrec
    hts}";
570 print"back to humans."
580 print"{7xrechts}{WIT}.{neer}
    {3xrechts}.{neer}{3xrechts}.
    {neer}{4xrechts}.{neer}{5xrechts
    }.":print"{op}{l.groen}{11xrecht
    s}machines."
590 print"{neer}{BLAUW}{11xrechts}
    machines."
600 print"{neer}{GEEL}{11xrechts}
    machines."
610 return
630 le=0:goto390

*** EINDE LISTING ***
```

öREGEL 0	248	REGEL 140	96
REGEL 10	83	REGEL 150	183
REGEL 20	171	REGEL 160	186
REGEL 30	110	REGEL 170	238
REGEL 35	192	REGEL 180	126
REGEL 40	172	REGEL 190	22
REGEL 50	73	REGEL 200	233
REGEL 60	185	REGEL 210	252
REGEL 70	16	REGEL 220	55
REGEL 80	11	REGEL 230	207
REGEL 90	221	REGEL 240	51
REGEL 100	166	REGEL 250	237
REGEL 110	193	REGEL 260	142
REGEL 120	183	REGEL 270	140
REGEL 130	248	REGEL 280	238

```
REGEL 285   127      REGEL 470   7
REGEL 290   80       REGEL 480   234
REGEL 300   17       REGEL 490   205
REGEL 320   97       REGEL 500   233
REGEL 330   180      REGEL 510   96
REGEL 340   101      REGEL 520   142
REGEL 350   49       REGEL 530   193
REGEL 360   176      REGEL 540   130
REGEL 361   60       REGEL 550   56
REGEL 365   101      REGEL 560   168
REGEL 370   158      REGEL 570   139
REGEL 380   206      REGEL 580   128
REGEL 390   123      REGEL 590   194
REGEL 400   7        REGEL 600   65
REGEL 410   37       REGEL 610   142
REGEL 420   44       REGEL 630   210
REGEL 440   255
REGEL 450   17
REGEL 460   166      READY.
```

Torenspel

Dit spel van Fons Reijsbergen is een computer-uitvoering van de toren van Babylon. De bedoeling is het vlak van 6x6 gekleurde cijfers in de juiste volgorde te krijgen. De 1 moet boven aan de rij staan. De volgorde van de kleuren is niet belangrijk maar iedere kolom moet zijn eigen kleur hebben. Je kunt een rij, een vakje draaien of een leeg vakje verplaatsen.
Een prima spel, we hadden ook niet anders verwacht van een van onze vaste inzenders.
Een waarschuwing vooraf: het spel is moeilijker dan het lijkt!

```
  1 rem toren spel / cbm 64
  2 rem door fons reijsbergen
  3 rem leidschendam / 070-27861
    9
  4 rem
 10 rem*******set up
 20 def fnr(x)=int(rnd(1)*x)+1
 30 gosub560
 40 z=0:w=0:dimb(6,7),o$(3000)
 45 i$="{HOME}{21xneer}":t$="{HOME}
    {6xneer}{5xrechts}":k$="{grijs 3
    }{l. rood}{l. groen}{l. blauw}{GROE
    N}{bruin}{3xZWART}"
 50 fori=1to6:forj=1to6:b(i,j)=i
    *10+j:nextj,i
 60 fori=1to6:b(i,7)=77:next:b(2
    ,7)=99:b(5,7)=99
 70 kl$="{grijs 3}{l. rood}{l. groen}
    {l. blauw}{GROEN}{bruin}{3xZWART}"

 74 rem**~~~~~~~~~~~~***************
    ********
 75 rem**123456789 dit zijn de k
    leuren**          **van de kolo
    men.              **
 76 rem**de kleuren van kl$ 678
    moeten**          **achtergrond
     kleur zijn.      **
 77 rem****************************
    ********
 80 gosub420
 90 rem*******input
100 printi$:input"Wat ga je doen
     :{9xspatie}{9xlinks}";a$:print
    tab(16)"{op}Momentje"
110 ifa$="w"thengosub390
120 ifa$="bs"thenrun
130 ifa$="bl"then350
140 iflen(a$)<>3then100
150 ifmid$(a$,2,1)<"0"ormid$(a$,
    2,1)>"7"then100
160 ifleft$(a$,1)="s"thengosub20
    0
170 ifleft$(a$,1)="d"thengosub29
    0
180 goto100
190 rem******schuif kolom op/nee
    r
200 r=val(mid$(a$,2,1)):k=asc(right$
    (a$,1))-64
210 l=6:ifk=2ork=5thenl=7
220 ifk<1ork>6or(l=6andr=7)thenreturn

230 r1=r:fori=1tol:ifb(k,i)=99then
    r1=i:i=l
240 next:o$="s"+right$(str$(r1),
    1)+right$(a$,1)
250 if(r1=r)or(k=5andr1=7andb(2,
    7)<>99)or(k=2andr1=7andb(5,7
    )<>99)thenreturn
260 r2=r-1:r3=1:ifr<r1thenr2=r+1
    :r3=-1
270 fori=r1tor2stepr3:b(k,i)=b(k
    ,i+r3):next:b(k,r)=99:goto51
    0
280 rem******schuif rij links/re
    chts
290 r=val(mid$(a$,2,1)):r$=right$
    (a$,1):ifr>6thenreturn
300 ifr$="l"theno=b(1,r):fori=1to
    5:b(i,r)=b(i+1,r):next:b(6,r
    )=o:o1$="r"
310 ifr$="r"theno=b(6,r):fori=6to
    2step-1:b(i,r)=b(i-1,r):next
    :b(1,r)=o:o1$="l"
320 ifr$="l"orr$="r"theno$="d"+mid$
    (a$,2,1)+o1$:goto510
330 return
340 rem*****terug draaien
350 w=1:forq=ztolstep-1:a$=o$(q)
    :ifleft$(a$,1)="s"thengosub2
    00:goto370
360 gosub290
370 next:w=0:z=0:goto100
380 rem*****in de war maken
390 fori=1to50:x=fnr(6):y=fnr(6)
    :x1=fnr(6):y1=fnr(6):o=b(x,y
    ):b(x,y)=b(x1,y1)
400 b(x1,y1)=o:next:goto520
410 rem*****speelveld tekenen
420 print"{CLR-HOME}{2xneer}rij{5xsp
    atie}kolom"tab(15)"{2xop}TOR
    EN SPEL{3xneer}":printtab(5)
    "A B C D E F"
430 fori=1to7:print"{neer}";i:next
    :print"{HOME}{4xneer}"
```

```
440 printtab(4)"0@2@2@2@2@2@.":for
    i=1to11:printtab(4)"] ] ] ]
    ] ] ]":next
450 printtab(4)"-@3 +@1@3 +@=":print
    tab(6)"] ]{3xspatie}] ]":print
    tab(6)"-@={3xspatie}-@={HOME}"

460 printtab(22)"{5xneer}BEWEGIN
    G":printtab(22)"{neer}Schuif
     rij{2xspatie}: d31"
470 printtab(22)"{neer}Schuif Ko
    lom: s4d":printtab(22)"{2xneer}
    BEGINSTAND"
480 printtab(22)"{neer}Snel{8xspatie
    }: bs":printtab(22)"{neer}La
    ngzaam{4xspatie}: bl"
490 printtab(22)"{neer}In de war
    {3xspatie}: w"
500 rem******ballen tekenen
510 ifw=0thenz=z+1:ifz<3001theno
    $(z)=o$
520 printt$;:fori=1to7:forj=1to6
    :printmid$(kl$,int(b(j,i)/10
    ),1);
530 print"{RVS-aan}";mid$(str$(b
    (j,i)),3,1);"{rechts}";:next
    j:print"{neer}":printtab(5):
    next
540 printtab(25)"{neer}{WIT}ZET
    : "z"{2xneer}":return
550 rem******spel regels
560 poke53280,0:poke53281,0:poke
    53272,23:poke657,128
570 print"{GEEL}{CLR-HOME}{11xspatie
    }** TOREN SPEL **"
580 print"{WIT}{neer}De bedoelin
    g is het vlak van 6*6 ge-{4xspat
    ie}kleurde cijfers in de ";
590 print"juiste volgorde tekrij
    gen."
600 print"{neer}De volgorde is v
    an 1 tot 6 en de 1 staatbove
    naan."
610 print"De volgorde van de kle
    uren doet er niet toe zolang
     elke kolom maar ";
620 print"zijn eigen{3xspatie}kl
    eur heeft."
630 print"{neer}Je kunt een rij
    een vakje draaien of eenleeg
     vakje verplaatsen in ";
640 print"de kolom waar dat vakj
    e in staat."
650 print"{neer}-d4l = draai rij
     4 links om"
660 print"-d2r = draai rij 2 rec
    hts om"
670 print"{neer}-s5d = schuif he
    t lege vakje in kolom d{8xspatie
    }naar rij 5"
680 gosub760:print"{GEEL}{11xspatie}
    ** TOREN SPEL **"
690 print"{neer}{WIT}De twee leg
    e vakjes onderaan kun je nie
    ttegelijk gebruiken."
700 print"{neer}Als je er niet u
    itkomt dan kun je het{3xspatie}
    veld weer goed zetten met:"
710 print"{neer}-bs = begin het
    programma opnieuw"
720 print"-bl = De computer herh
    aalt al je zetten{7xspatie}i
    n omgekeerde volgorde."
730 print"{neer}-w{2xspatie}= go
    oi alles door elkaar.{17xspatie}
    (bl zet alles in de ";
740 print"nu verkregen{9xspatie}
    volgorde )
750 print"{2xneer}Sterkte....."
    :goto760
760 print"{2xneer}{GROEN}{10xspatie}
    DRUK OP EEN TOETS{WIT}"
770 geta$:ifa$=""then770
780 print"{CLR-HOME}";:return
```

*** EINDE LISTING ***

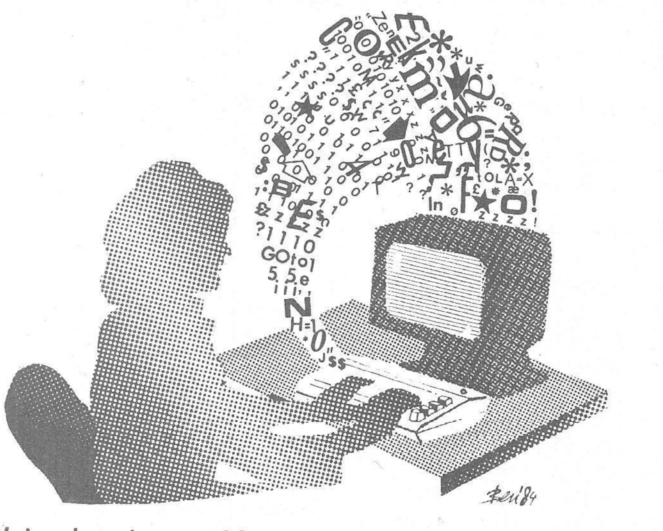
torenspel

REGEL 1	182	REGEL 360	40
REGEL 2	41	REGEL 370	191
REGEL 3	36	REGEL 380	1
REGEL 4	143	REGEL 390	251
REGEL 10	70	REGEL 400	233
REGEL 20	10	REGEL 410	57
REGEL 30	40	REGEL 420	173
REGEL 40	78	REGEL 430	94
REGEL 45	243	REGEL 440	0
REGEL 50	76	REGEL 450	193
REGEL 60	235	REGEL 460	38
REGEL 70	120	REGEL 470	47
REGEL 74	205	REGEL 480	138
REGEL 75	40	REGEL 490	57
REGEL 76	168	REGEL 500	67
REGEL 77	249	REGEL 510	44
REGEL 80	35	REGEL 520	163
REGEL 90	69	REGEL 530	253
REGEL 100	203	REGEL 540	91
REGEL 110	13	REGEL 550	129
REGEL 120	172	REGEL 560	196
REGEL 130	179	REGEL 570	242
REGEL 140	211	REGEL 580	209
REGEL 150	60	REGEL 590	228
REGEL 160	117	REGEL 600	122
REGEL 170	111	REGEL 610	113
REGEL 180	26	REGEL 620	157
REGEL 190	199	REGEL 630	75
REGEL 200	148	REGEL 640	159
REGEL 210	230	REGEL 650	179
REGEL 220	240	REGEL 660	236
REGEL 230	64	REGEL 670	238
REGEL 240	183	REGEL 680	195
REGEL 250	142	REGEL 690	32
REGEL 260	170	REGEL 700	121
REGEL 270	229	REGEL 710	0
REGEL 280	171	REGEL 720	139
REGEL 290	198	REGEL 730	213
REGEL 300	204	REGEL 740	146
REGEL 310	84	REGEL 750	21
REGEL 320	145	REGEL 760	94
REGEL 330	142	REGEL 770	107
REGEL 340	6	REGEL 780	115
REGEL 350	30		

READY.

Typo Temper

Dit is een programma, waarbij de typesnelheid op de proef wordt gesteld, een beetje in de lijn van Tempo Typen van Radarsoft.Rechts in beeld verschijnt een woord, dat U moet intypen, voordat het de barriere bereikt. Er zijn 5 rondes van 5 woorden. In het begin valt het nog wel mee, maar uw tijd wordt steeds korter. Het spel is uit, als u alle 5 rondes hebt overleefd, of als U 4 keer gefaald hebt, om het woord op tijd in te tikken. U kunt dan kiezen, of U nog een spel wil doen.



Wat gebeurt er zoal in het programma?

Nadat in regel 1010 de schermkleur bruin is geworden, komen we al direkt bij het belangrijkste deel van het programma.In de regels 1020-1100 worden de gegevens, waarmee de C-64 zijn karakterset opmaakt voor de letters van het alfabet, gebruikt om sprite-tekeningen mee te maken. Nu bstaat een sprite-tekening uit een matrix van 24 bij 21 beeldpunten, terwijl een gewoon karakter maar 8 bij 8 beeldpunten telt. En omdat de geheugenplaatsen waar ik de sprite-tekeningen wil gaan vormen, normaal altijd al zijn gevuld met rommel, moeten ze eerst schoongepoetst worden.

Dit gebeurt in 1040, waar de blokken 229 t/m 255 worden schoongemaakt. Blok 229 blijft dan gereserveerd voor een spatie(blanko), in de hogere blokken komen de 26 letters van het alfabet. De regels 1020,1030,1090 en 1100 roepen de standaard karakterset van de Commodore op en gebruiken deze voor eigen doeleinden. Laten we de gecompliceerde dubbele for.next.lus in 1050-1080 eens goed bekijken.

Laten we aannemen, dat we de lus voor de eerste keer doorlopen. Dus P=1. In 1060 lezen we dan:
"PG(de plaats waar we iets gaan poken)=14656 + 1 * 64=14720." Dit is het beginadres van sprite-blok 230 (230 * 64 = 14720). Vervolgens: "LG" plaats waar we een getal gaan lezen)= 53248 + 1 * 8 = 53256." Dit is de plaats waar de karakterset van de letter A is opgeslagen. Terwijl P nog steeds 1 is, gaan we een nieuwe for.next introduceren, waarbij Q een waarde kan krijgen van 0 tot 7.

Laten we aannemen dat Q de waarde van nul heeft, dan wordt regel 1080: poke op adres 14720 het getal, dat nu geschreven staat op adres 53256". Bij Q=1 wordt dit 14723 resp. 53257, waarna op identieke wijze de gegevens van de letter A worden gecopieerd naar sprite-blok 230. Als Q=7, dan is deze for.next lus afgewerkt, en krijgt P de waarde van 2. Op deze manier worden 26 spritetekeningen gevormd, door de karakterset over te nemen. Als men over de hond komt, dan komt men ook over de staart. De rest van het programma is niet zo gecompliceerd. In 1120 wordt een woordenschat van 50 woorden min of meer in geheimtaal weggeschreven, ergens in het grote geheugen. Het zou natuurlijk wel mogelijk zijn, om dit te doen in de vorm van "strings", maar dan kan men er, door naar de listing te kijken, heel makkelijk achterkomen dat het eerste woord "toetsen" is.

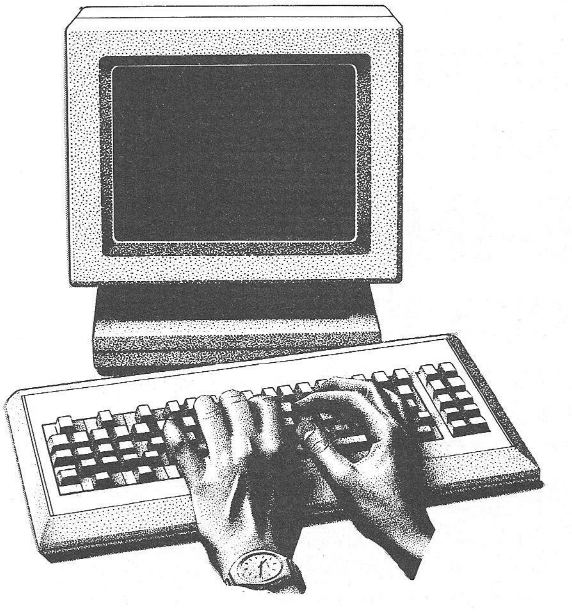
Als men nu "vals" wil spelen, dan moet men eerst de vijftig data-regels helemaal gaan ontleden volgens de formule in regel 1110. In 1140 wordt de geluids-chip gereset (alles op 0 gesteld), waarna voorlopig alleen de "sustain-release" en het volume worden aangegeven. Daarna verschijnt een korte instruktie, en start het spel in 1230. De ronde (ro) is dan 0, de score eveneens (SC); de rand krijgt een groene kleur en het aantal missers dat men mag maken (AM) wordt 4. Vervolgens wordt het scherm geprint.

Omdat men meerdere spelletjes achter elkaar kan doen, kan men ook een topscore (TS) hebben. Hieronder komt een mooi kader, waarbinnen het spel zich gaat afspelen. In 1290 wordt het aantal missers gesymboliseerd. Het teken dat hier geprint wordt, is het klavertje; op mijn printer wordt dit welhaast onherkenbaar weergegeven.

Om het plaatje wat te verfraaien, worden er 10 sterren in gepoked. Hiervoor worden een horizontale plaats (HP), een vertikale (VP) en een totale plaats (TP) berekend, waarna het karakter "inversed" wordt gepoked met kleur grijs 2. In 1330 krijgen alle sprites een witte kleur, en er wordt alvast voor gezorgd, dat ze straks allemaal rechts in beeld zullen verschijnen. Als in 1350 een ronde start, dan is om te beginnen het aantal woorden AW=0. De rondteller wordt opgehoogd, en op de tweede regel geprint. Daarna wordt de barriere geplaatst. De plaats ervan (BP) is afhankelijk van de ronde, en schuift iedere ronde dus 1 positie naar rechts. Eerst wordt een vertikale kolom gevuld met blokjes grijs 2, zodat de barriere van een voorgaande ronde wordt overschreven. Daarna komt dan de eigenlijke barriere in de vorm van lichtgroene blokjes. We moeten ook nog bepalen, wanneer een woord met de barriere in botsing zal komen; dit noem ik ML (maximaal links), en dit is ook afhankelijk van de ronde. Vervolgens moeten we een woord kiezen. Het is dan duidelijk, dat we nadien een eerste letter zullen moeten intikken (IL). We hebben dan de keus uit 50 woorden. Het aantal woorden in deze ronde wordt met 1 verhoogd. We halen eventuele letters weg, die er nog stonden als het vorige woord mislukt was (1440). Aan de hand van de gedane woordkeus WK wordt het begingetal bepaald, om het in kode opgeslagen woord te bepalen. Als bijvoorbeeld WK=1, dan is dus het begin getal 20010, hetgeen eerst het aantal letters oplevert (1460). In 1470 krijgen de acht sprites de verwijzingen naar het te gebruiken teken-blok. In het geval van ons woord "toetsen" krijgt dus de eerste sprite (2039 + 1) het blok 20 + 229 = 249 toegekend, waaruit de letter "t" voortkomt. Vervolgens worden in 1480-1550 acht strings gevormd aan de hand van de benodigde

letters, waarna we eindelijk het woord in beeld kunnen brengen. Eerst bepaalt de computer de vertikale plaats VP, welke een spreiding heeft van 60 pixels. De gevonden waarde wordt toegekend aan alle acht sprites(1570). Vervolgens krijgen ze allemaal een horizontale plaats (HA, HB etc), die per letter steeds 8 pixels hoger is. Nu dit alles gereed is, kunnen de sprites in beeld worden gebracht, en staat het woord dus gedeeltelijk zichtbaar op het scherm(1600). De snelheid, waarmee het woord gaat schuiven, is weer afhankelijk van de ronde (SS in 1610); er wordt een toonhoogte ingezet voor stem 1(1620), waarna het spel kan beginnen.

De eigenlijke programma-lus is in wezen heel simpel.In regel 160 wordt gekeken, welke toets er is ingedrukt. Afhankelijk van de in te tikken letter, wordt naar het desbetreffende blokje gesprongen. Wanneer de letter in dit blok de barriere bereikt heeft (als HA kleiner is dan de in



1400 berekende waarde van ML), dan heeft men dus het woord niet op tijd ingetikt, en gaan we naar 800. Laten we eerst even kijken, wat er gebeurt, als we gefaald hebben. Uw score wordt verlaagd met het aantal letters van het woord, vermenigvuldigd met de ronde, waarin U speelde. De nieuwe score wordt op een schoongemaakte regel (zie 820) geprint (zie 830). De stem wordt uitgezet, de randkleur wordt rood(840); het aantal missers wordt bijgewerkt, en er wordt een klavertje weggepoked(850). Na een kort wachtlusje wordt de stem weer aangezet met een aflopende toonhoogte, waarna de stem weer uitgaat, en de rand weer groen wordt.
Als we nog geen vier missers gehad hebben, en dus AM nog groter is dan nul, dan mogen we het nog eens proberen. Aangezien het woord mislukt is, moet het aantal woorden verlaagd worden (900), waarna een nieuw woord getrokken wordt in 1410.
Als het wel de vierde misser was, dan is het spel uit (910). Er wordt een kort muziekje gespeeld (930-950), waarna gevraagd wordt, of U nog een spel wilt doen. Zo ja, dan gaan we naar 1230, zo nee, dan wordt (met een in uw C-64 ingebakken sys-commando) uw computer gereset (hetzelfde zou U bereiken door runstop/restore in te tikken), waarbij overigens het programma in het geheugen blijft.
We keren nu terug naar de lus, en volgen deze, als het goed gaat. Wanneer de ingetikte letter gelijk is aan bijvoorbeeld de eerste letter van het woord(190), dan wordt het tik-geluid aangezet, de eerste letter wordt weggehaald, en IL wordt verhoogd (210), zodat de volgende keer niet meer naar 180, maar naar 220 wordt gesprongen. Als de ingetikte letter niet overeenkomt (of als er niets was ingetikt), dan gaan we naar 540, waar de letters een voor een verplaatst worden.
Op soortgelijke wijze is voor alle acht letters een blokje gecreëerd, met voor zover nodig aangepaste gegevens. Want als men bijvoorbeeld drie letters heeft ingetikt, dan mogen deze niet meer verplaatst worden, dus gaat men vanuit 320 rechtstreeks naar 570. Vanaf de vijfde letter wordt gecontroleerd, of men nu het hele woord heeft ingetikt. (in mijn 50 woorden zijn ze allemaal vijf of meer letters).
Indien dit niet het geval is, dan gaat men weer verder in de lus. Uiteindelijk komt men na het verplaatsen van de letters in 620 terecht, waar het eventuele tik-geluid weer wordt uitgezet, waarna men teruggaat naar het begin van de lus. Vanaf 630 zien we, wat er gebeurt, als men wel het hele woord heeft ingetikt. De score wordt verhoogd met het aantal letters van het woord, vermenigvuldigd met de ronde. Als de score hiermee hoger is geworden dan de topscore, dan wordt de topscore bijgewerkt. De bovenste schermregel wordt (gedeeltelijk) gewist, waarna score en topscore worden bijgewerkt. Het tikgeluid wordt uitgezet (670) waarna gekeken wordt, of we al vijf woorden in deze ronde hebben ingetikt. Zo niet, dan trekken we een nieuw woord in 1410. Als het wel het einde van de ronde is, dan wordt de rand paars, en speelt een klein muziekje (690-740).

Als we nog niet de vijfde ronde hadden afgewerkt (760), dan moet er nog een nieuwe ronde volgen. Als het wel de vijfde ronde was, dan wordt het muziekje beeindigd (770-780), waarna het hele spel rond is, en men kan intoetsen, of men het nog eens wil proberen.

Wijzigingen in het programma
Men hoeft geen doorgewinterde programmeur te zijn, om diverse aanpassingen in dit programma te plegen. De structuur is natuurlijk ook zeer duidelijk te volgen aan de hand van het bovenstaande en er worden zo weinig mogelijk variabelen gebruik. Zo wordt de randkleur nogal vaak veranderd (poke 43280), maar ik gebruik hier geen variabele voor. Hierdoor wordt het programma op zich wel wat langer, maar het heeft het voordeel, dat men in iedere regel kan zien, wat er gebeurt. Men kan andere woorden er inzetten.
Per data-regel moet men opgeven: het aantal letters, de betreffende letters van het alfabet, aangevuld met nullen, tot men op deze regel tien cijfers heeft. Het is noodzakelijk, dat een woord minstens vijf letters heeft, anders moet men in de programmalus invoegen een regel 335: if IL = AL + 1 then 630. Men kan de woordenschat verkleinen c.q. vergroten door de read/poke in 1120 en de formule voor de woord-keuze in 1430 aan te passen en natuurlijk het aantal data-regels te verkleinen c.q. vergroten. Men

kan het aantal woorden per ronde heel eenvoudig wijzigen in de formule op het eind van regel 670.
Het aantal ronden kan verkleind of vergroot worden (een zesde ronde onder dezelfde condities is praktisch onspeelbaar) door aanpassing van regel 760.Tenslotte kan men de snelheid, waarmee het woord naar links schuift, aanpassen in regel 1610.

Conclusie.
Hopelijk hebt U enige wijsheid kunnen putten uit de toelichting bij de opbouw van het programma. U zou bijvoorbeeld de truc met de letter-sprites kunnen gebruiken om op een mooie manier tekst op een scherm te laten verschijnen; U kunt ook de letters (of andere karakters) vergroten. Aan de hand van de tips, om het programma naar uw eigen inzicht te veranderen kan ook een beginnend programmeur aan het werk, om het aan te passen aan de persoonlijke smaak.

```
100 rem *** tempo-typen
110 rem *** door wim stadhouders
120 rem *** scheldekade 2
130 rem *** 4531 ee  terneuzen
140 goto1000
150 rem lus
160 getz$
170 on il goto180,220,260,300,34
    0,390,440,490
180 if ha<ml then800
190 ifz$=a$then210
200 goto540
210 poke54276,129:poke53269,254:
    il=2
220 if hb<ml then800
230 ifz$=b$then250
240 goto550
250 poke54276,129:poke53269,252:
    il=3
260 if hc<ml then800
270 ifz$=c$then290
280 goto560
290 poke54276,129:poke53269,248:
    il=4
300 if hd<ml then800
310 ifz$=d$then330
320 goto570
330 poke54276,129:poke53269,240:
    il=5
340 if he<ml then800
350 ifz$=e$then370
360 goto580
370 poke54276,129:poke53269,224:
    il=6
380 if il=al+1then630
390 if hf<ml then800
400 ifz$=f$then420
410 goto590
420 poke54276,129:poke53269,192:
    il=7
430 if il=al+1then630
440 if hg<ml then800
450 ifz$=g$then470
460 goto600
470 poke54276,129:poke53269,128:
    il=8
480 if il=al+1then630
490 if hh<ml then800
500 ifz$=h$then520
510 goto610
520 poke54276,129:poke53269,0:goto
    630
530 rem verplaatsen
540 ha=ha-ss:poke53248,ha
550 hb=hb-ss:poke53250,hb
560 hc=hc-ss:poke53252,hc
570 hd=hd-ss:poke53254,hd
580 he=he-ss:poke53256,he
590 hf=hf-ss:poke53258,hf
600 hg=hg-ss:poke53260,hg
610 hh=hh-ss:poke53262,hh
620 poke54276,128:goto150
630 rem woord weg
640 sc=sc+(al*ro):ifsc>ts then t
    s=sc
650 forp=1024to1044:pokep,32:next
660 print"{HOME}{GEEL}score"sc"{3xsp
    atie}topscore"ts
670 poke54276,128:ifaw<5then1410
680 forp=1to500:next:poke53280,4
690 poke54273,18:poke54272,209:poke
    54276,33:forp=1to300:next
700 poke54273,14:poke54272,24:for
    p=1to600:next
710 poke54273,18:poke54272,209:for
    p=1to300:next
720 poke54273,23:poke54272,181:for
    p=1to300:next
730 poke54273,28:poke54272,48:for
    p=1to300:next
740 poke54276,32:forp=1to50:next
750 poke53280,5
760 if ro<5then1340
770 poke54276,33:forp=1to250:next
780 poke54273,37:poke54272,162:for
    p=1to1000:next:poke54276,32
790 print"{HOME}{2xneer}{WIT}goe
    d gedaan jochie":goto960
800 rem verloren
810 sc=sc-(al*ro):ifsc<0then sc=
    0
820 forp=1024to1044:pokep,32:next
830 print"{HOME}{GEEL}score"sc"{3xsp
    atie}topscore"ts
840 poke54276,128:poke53280,2
850 am=am-1:poke1704+am,32
860 forp=1to500:next
870 poke54273,80:poke54276,17
880 forp=80to0step-.5:poke54273,
    p:next:poke54276,16:forp=1to
    500:next
890 poke53280,5
900 if am>0thenaw=aw-1:goto1410
910 rem einde spel
920 print"{HOME}{2xneer}{WIT}ein
    de spel...... "
930 poke54273,16:poke54272,195:poke
    54276,17:forp=1to800:next
940 poke54273,18:forp=1to800:next
    :poke54273,16:forp=1to800:next
950 poke54273,19:forp=1to1600:next
    :poke54276,16
```

```
 960 forp=1to1000:next:print"{WIT}
     <<<< nog een spel (j/n)? >>>
     >"
 970 geta$:ifa$="n"thensys65126
 980 ifa$="j"then1230
 990 goto970
1000 rem spriteblokken
1010 poke53281,9:print"{CLR-HOME}
     {GEEL}dit duurt wel even...!"
1020 poke56334,peek(56334)and254
1030 poke1,peek(1)and251
1040 forp=14656to16384:pokep,0:next
1050 forp=1to26
1060 pg=14656+p*64:lg=53248+p*8
1070 forq=0to7
1080 pokepg+q*3,peek(lg+q):next:next
1090 poke1,peek(1)or4
1100 poke56334,peek(56334)or1
1110 rem woorden:aantal letters/l
     etter van alfabet/dummy's
1120 forp=20010to20509:reada:poke
     p,a:next
1130 rem instruktie
1140 forp=54272to54296:pokep,0:next
     :poke54278,72:poke54296,15
1150 print"{CLR-HOME}{neer}{8xspatie}
     < tempo - typen >"
1160 print"{neer}rechts in beeld
     verschijnt een woord."
1170 print"u dient dit woord in t
     e typen, voordat"
1180 print"het de barriere bereik
     t heeft."
1190 print"{neer}er zijn 5 rondes
      van elk 5 woorden,"
1200 print"waarbij uw tijd steeds
      korter wordt."
1210 print"{3xneer}<< druk op een
      toets om te beginnen. >>"
1220 getw$:ifw$=""then1220
1230 rem start spel
1240 ro=0:sc=0:poke53280,5:am=4
1250 print"{CLR-HOME}{GEEL}score"
     sc"{3xspatie}topscore"ts
1260 print"{4xneer}{CYAAN}{RVS-aan}
     [[[[[[[[[[[[[[[[[[[[[[[[[[[[
     [[[[[[[[[[[[";
1270 forp=1to9:print"{grijs 2}{RVS-aa
     n}{30xspatie}{grijs 1}{10xspatie
     }";:next
1280 print"{CYAAN}{RVS-aan}[[[[[[
     [[[[[[[[[[[[[[[[[[[[[[[[[[[[
     [[[[[[";
1290 print"{neer}{WIT}{RVS-uit}XX
     XX"
1300 rem sterren
1310 forp=1to10:hp=int(28*rnd(1))
     :vp=int(7*rnd(1)):tp=1304+hp
     +(vp*40)
1320 poketp+54272,12:poketp,170:next
1330 forp=53287to53294:pokep,1:next
     :poke53264,255
1340 rem start ronde
1350 aw=0:ro=ro+1:print"{HOME}{neer}
     {GEEL}ronde"ro
1360 rem barriere
1370 bp=1292+ro
1380 forp=bp to bp+320step40:poke
     p+54272,12:pokep,160:next:bp
     =bp+1
1390 forp=bp to bp+320step40:poke
     p+54272,13:pokep,127:next
1400 ml=8+(ro*8):rem maximaal lin
     ks
1410 rem woordkeus
1420 il=1:rem in te tikken letter
1430 wk=int(50*rnd(1))+1:aw=aw+1
1440 poke53269,0
1450 bg=20000+(wk*10)
1460 al=peek(bg):rem aantal lette
     rs
1470 forp=1to8:poke2039+p,peek(bg
     +p)+229:next
1480 a$=chr$(peek(bg+1)+64)
1490 b$=chr$(peek(bg+2)+64)
1500 c$=chr$(peek(bg+3)+64)
1510 d$=chr$(peek(bg+4)+64)
1520 e$=chr$(peek(bg+5)+64)
1530 f$=chr$(peek(bg+6)+64)
1540 g$=chr$(peek(bg+7)+64)
1550 h$=chr$(peek(bg+8)+64)
1560 rem sprites aan
1570 vp=int(60*rnd(1))+100:forp=5
     3249to53263step2:pokep,vp:next
1580 ha=70:poke53248,ha:hb=78:poke
     53250,hb:hc=86:poke53252,hc
1590 hd=94:poke53254,hd:he=102:poke
     53256,he:hf=110:poke53258,hf
1600 hg=118:poke53260,hg:hh=126:poke
     53262,hh:poke53269,255
1610 ss=.7+(ro*.3):rem snelheid s
     prites
1620 poke54273,20:goto150
1630 data7,20,15,5,20,19,5,14,0,0
1640 data8,3,15,13,16,21,20,5,18,
     0
1650 data8,18,5,3,15,18,4,5,18,0
1660 data8,7,5,19,3,8,9,11,20,0
1670 data5,18,1,4,9,15,0,0,0,0
1680 data5,20,25,16,5,14,0,0,0,0
1690 data7,20,25,16,5,12,5,19,0,0
1700 data8,11,1,16,9,20,5,9,14,0
1710 data8,15,16,4,18,1,3,8,20,0
1720 data8,15,16,4,18,1,7,5,14,0
1730 data7,11,12,5,21,18,5,14,0,0
1740 data6,2,21,9,7,5,14,0,0,0
1750 data8,2,21,9,7,26,1,13,5,0
1760 data5,12,21,9,5,18,0,0,0,0
1770 data7,12,21,9,5,18,5,14,0,0
1780 data8,1,17,21,1,4,21,3,20,0
1790 data8,1,17,21,1,12,15,14,7,0
1800 data7,3,5,14,20,18,21,13,0,0
1810 data8,3,5,14,20,18,1,12,5,0
1820 data8,3,8,18,9,19,20,21,19,0
1830 data5,5,24,20,18,1,0,0,0,0
1840 data7,5,24,20,18,1,3,20,0,0
1850 data6,5,24,16,5,18,20,0,0,0
1860 data6,5,24,16,15,18,20,0,0,0
1870 data7,10,5,18,5,13,9,1,0,0
1880 data6,16,1,21,12,21,19,0,0,0
```

```
1890 data8,15,16,5,14,2,1,18,5,0
1900 data6,7,5,8,5,9,13,0,0,0
1910 data7,2,12,9,10,11,5,14,0,0
1920 data5,11,12,5,9,14,0,0,0,0
1930 data8,16,15,19,9,20,9,5,6,0
1940 data7,16,15,19,9,20,9,5,0,0
1950 data6,7,5,12,9,10,11,0,0,0
1960 data8,7,5,12,9,10,11,5,14,0
1970 data7,7,5,12,15,6,20,5,0,0
1980 data7,7,5,12,15,22,9,7,0,0
1990 data7,19,20,9,10,7,5,14,0,0
2000 data7,19,20,5,9,7,5,18,0,0
2010 data6,11,18,1,20,5,18,0,0,0
2020 data5,11,1,20,5,18,0,0,0,0
2030 data8,1,13,1,26,15,14,5,19,0
2040 data6,13,5,24,9,3,15,0,0,0
2050 data6,2,5,12,7,9,5,0,0,0
2060 data6,20,8,1,13,5,19,0,0,0
2070 data8,22,12,9,5,12,1,14,4,0
2080 data6,9,19,18,1,5,12,0,0,0
2090 data7,12,15,11,1,20,9,5,0,0
2100 data7,16,18,9,13,1,9,18,0,0
2110 data8,2,18,15,5,11,26,1,11,0
2120 data8,22,9,10,6,20,9,5,14,0
*** EINDE LISTING ***
```

typo temper

```
REGEL 100   79       REGEL 470   193
REGEL 110   116      REGEL 480   122
REGEL 120   76       REGEL 490   166
REGEL 130   36       REGEL 500   101
REGEL 140   74       REGEL 510   32
REGEL 150   131      REGEL 520   249
REGEL 160   31       REGEL 530   212
REGEL 170   163      REGEL 540   161
REGEL 180   159      REGEL 550   157
REGEL 190   90       REGEL 560   162
REGEL 200   34       REGEL 570   167
REGEL 210   187      REGEL 580   172
REGEL 220   160      REGEL 590   177
REGEL 230   95       REGEL 600   173
REGEL 240   35       REGEL 610   178
REGEL 250   186      REGEL 620   191
REGEL 260   161      REGEL 630   253
REGEL 270   100      REGEL 640   252
REGEL 280   36       REGEL 650   37
REGEL 290   192      REGEL 660   250
REGEL 300   162      REGEL 670   24
REGEL 310   96       REGEL 680   220
REGEL 320   37       REGEL 690   25
REGEL 330   185      REGEL 700   120
REGEL 340   163      REGEL 710   174
REGEL 350   101      REGEL 720   169
REGEL 360   38       REGEL 730   128
REGEL 370   188      REGEL 740   227
REGEL 380   122      REGEL 750   250
REGEL 390   164      REGEL 760   131
REGEL 400   98       REGEL 770   22
REGEL 410   39       REGEL 780   69
REGEL 420   193      REGEL 790   234
REGEL 430   122      REGEL 800   252
REGEL 440   165      REGEL 810   17
REGEL 450   104      REGEL 820   37
REGEL 460   31       REGEL 830   250
```

```
REGEL 840   151      REGEL 1500  190
REGEL 850   16       REGEL 1510  192
REGEL 860   169      REGEL 1520  194
REGEL 870   157      REGEL 1530  196
REGEL 880   211      REGEL 1540  198
REGEL 890   250      REGEL 1550  200
REGEL 900   232      REGEL 1560  137
REGEL 910   40       REGEL 1570  221
REGEL 920   242      REGEL 1580  7
REGEL 930   34       REGEL 1590  112
REGEL 940   102      REGEL 1600  173
REGEL 950   179      REGEL 1610  165
REGEL 960   32       REGEL 1620  131
REGEL 970   189      REGEL 1630  9
REGEL 980   157      REGEL 1640  62
REGEL 990   41       REGEL 1650  233
REGEL 1000  108      REGEL 1660  181
REGEL 1010  125      REGEL 1670  113
REGEL 1020  42       REGEL 1680  206
REGEL 1030  127      REGEL 1690  13
REGEL 1040  108      REGEL 1700  223
REGEL 1050  192      REGEL 1710  223
REGEL 1060  194      REGEL 1720  227
REGEL 1070  143      REGEL 1730  6
REGEL 1080  77       REGEL 1740  116
REGEL 1090  28       REGEL 1750  174
REGEL 1100  193      REGEL 1760  161
REGEL 1110  195      REGEL 1770  221
REGEL 1120  102      REGEL 1780  208
REGEL 1130  161      REGEL 1790  9
REGEL 1140  75       REGEL 1800  5
REGEL 1150  61       REGEL 1810  216
REGEL 1160  59       REGEL 1820  29
REGEL 1170  29       REGEL 1830  155
REGEL 1180  51       REGEL 1840  210
REGEL 1190  0        REGEL 1850  215
REGEL 1200  17       REGEL 1860  8
REGEL 1210  233      REGEL 1870  168
REGEL 1220  190      REGEL 1880  0
REGEL 1230  81       REGEL 1890  223
REGEL 1240  23       REGEL 1900  75
REGEL 1250  122      REGEL 1910  209
REGEL 1260  69       REGEL 1920  156
REGEL 1270  224      REGEL 1930  237
REGEL 1280  1        REGEL 1940  230
REGEL 1290  229      REGEL 1950  160
REGEL 1300  178      REGEL 1960  220
REGEL 1310  153      REGEL 1970  168
REGEL 1320  109      REGEL 1980  175
REGEL 1330  14       REGEL 1990  221
REGEL 1340  149      REGEL 2000  181
REGEL 1350  117      REGEL 2010  209
REGEL 1360  219      REGEL 2020  151
REGEL 1370  93       REGEL 2030  15
REGEL 1380  128      REGEL 2040  166
REGEL 1390  153      REGEL 2050  68
REGEL 1400  151      REGEL 2060  163
REGEL 1410  82       REGEL 2070  217
REGEL 1420  7        REGEL 2080  170
REGEL 1430  122      REGEL 2090  210
REGEL 1440  252      REGEL 2100  230
REGEL 1450  215      REGEL 2110  10
REGEL 1460  120      REGEL 2120  224
REGEL 1470  238
REGEL 1480  186
REGEL 1490  188      READY.
```

Omdat er veel vraag naar is herplaatsen wij in dit nummer de C-16 checksum.

Checksum C-16

```
 10 rem ******************************
    *****
 20 rem syntax.checksum
 30 rem voor c-16 & plus/4
 40 rem
 50 rem syntax testen met 'sys 1536'

 60 rem
 70 rem v.851128.16        jan bodzing
    a
 80 rem ******************************
    *****
 90 i=1536        :rem beginadres
100 reada:ifa>=0then pokei,a:i=i+1:goto
    100
110 print"data is weggezet"
120 print"cheksum printen met 'sys 1
    536'
130 end
200 data 165, 43,166, 44,133
210 data  31,134, 32,169,147
220 data  32,210,255,160,  0
230 data 240,  3, 32, 73,  6
240 data  32, 73,  6,208,  1
250 data  96, 72,152, 32,131
260 data   6,168,104,234, 32
270 data  81,  6, 32, 73,  6
280 data 240, 12,201, 32,240
290 data 247, 24,101,252,133
300 data 252, 76, 37,  6,166
310 data 252,169,  0,132,253
320 data  32, 95,164,169, 13
330 data  32,210,255,164,253
340 data  76, 17,  6,200,208
350 data   2,230, 32,177, 31
360 data  96,162,  0,189,123
370 data   6,240,  6, 32,210
380 data 255,232,208,245, 32
390 data  73,  6,170, 32, 73
400 data   6,132,253, 32, 95
410 data 164,162,  3,169, 32
420 data  32,210,255,202,208
430 data 250,169,  0,133,252
440 data 164,253, 96, 82, 69
450 data  71, 69, 76, 32,  0
460 data   0, 72,138, 72, 32
470 data 225,255,240,251,104
480 data 170,104, 96, -1

*** EINDE LISTING ***
```

8 Koninginnen C-16

Dit programma is ingezonden door Niek Meijer uit Maasland. Bij 8 Koninginnen is het de bedoeling de 8 Koninginnen zodanig op een schaakbord te zetten dat geen van deze Koninginnen geslagen kan worden. Zoals U waarschijnlijk weet beslaat een Koningin in het schaakspel de gehele rij, de kolom en de beide diagonalen waarop het stuk zich bevindt.

Na het opstarten kunt U met de cursor op en neer om het stuk verplaatsen en na een return wordt het stuk in de huidige positie geplaatst en verschijnt de volgende Koningin. Komt U er niet meer uit dan kunt U met de cursor-links toets weer een of meerdere kolommen terug.

Toets U echter (c) van de computer in dan zal de computer trachten, uitgaande van de huidige opstelling, een oplossing te vinden.

```
 2 dimd1(14),d2(14)
 4 fori=1to70:reada:K$=K$+chr$(
   a):next
 6 graphic1,1:box1,64,0,256,192

 8 x1=64:y1=24:fori=1to2
10 forx=x1to255step48:fory=y1to
   191step48
12 box,x,y,x+23,y+23,,1:nexty,x

14 x1=88:y1=0:nexti
16 fori=0to7:char,3*i+9,24,chr$
   (65+i),1:char,5,3*i+1,str$(8
   -i):next
18 K=0:r=0:m=1
20 gosub62:do:getKeyy$
22 ifinstr("{links}cs"+chr$(13)
   ,y$)<>0thenexit
24 ify$<>"{op}"andy$<>"{neer}"then
   32
26 gosub62:ify$="{op}"andr<7then
   r=r+1
28 ify$="{neer}"andr>0thenr=r-1

30 gosub62
32 loop
34 ify$=chr$(13)thengosub64
36 ify$="s"orm=0then56
38 ify$="{links}"thengosub72
40 ify$<>"c"then20
42 gosub62:K1=K:m=0:r1=0:dountil
   K>7orm=1
44 r=r1:p=0:dountilr>7orp=1
46 ifr(r)=0andd1(7+K-r)=0andd2(
   14-K-r)=0thenpK(K)=r:p=1:else
   r=r+1
48 loop
50 ifp=1thengosub82
52 ifp=0thengosub84
54 loop
56 ifm=1thenchar,1,1,"x ":elsechar
   ,1,1,"oK"
58 getKeyy$:K1=K:forK=0toK1-1:r
   =pK(K):gosub62:r(r)=0:d1(7+K
   -r)=0:d2(14-K-r)=0
60 next:char,1,1,"{2xspatie}":goto
   18
```

```
62 gshapek$,k*24+68,24*(7-r)+1,
   4:return
64 ifr(r)<>0ord1(7+k-r)<>0ord2(
   14-k-r)<>0thengosub62:return
66 pk(k)=r:r(r)=1:d1(7+k-r)=1:d
   2(14-k-r)=1
68 ifk<7thenk=k+1:r=0:return
70 m=0:return
72 ifk=0then80
74 gosub62:ifk<>0thenk=k-1:r=pk
   (k)
76 r(r)=0:d1(7+k-r)=0:d2(14-k-r
   )=0:ifk=0then80
78 fori=0tok-1:j=pk(i):r(j)=1:d
   1(7+i-j)=1:d2(14-i-j)=1:next
80 gosub62:return
82 pk(k)=r:r(r)=1:d1(7+k-r)=1:d
   2(14-k-r)=1:gosub62:k=k+1:r1
   =0:return
84 ifk>k1thengosub72:r1=pk(k)+1
   :return
86 ifk=k1andr>7thenm=1:return
88 data1,192,0,193,193,128,243,
   231,128,127,255,0,127,255,0,
   63,254,0,63,254,0
90 data31,252,0,31,252,0,15,248
   ,0,15,248,0,7,240,0,15,248,0
   ,15,248,0,3,224,0
92 data3,224,0,3,224,0,3,224,0,
   7,240,0,15,248,0,63,254,0,25
   5,255,128,16,0,21,0

*** EINDE LISTING ***
```

REGEL 2	9	REGEL 50	92
REGEL 4	67	REGEL 52	93
REGEL 6	59	REGEL 54	236
REGEL 8	62	REGEL 56	119
REGEL 10	13	REGEL 58	169
REGEL 12	51	REGEL 60	242
REGEL 14	86	REGEL 62	185
REGEL 16	60	REGEL 64	14
REGEL 18	5	REGEL 66	149
REGEL 20	107	REGEL 68	192
REGEL 22	30	REGEL 70	247
REGEL 24	50	REGEL 72	199
REGEL 26	129	REGEL 74	217
REGEL 28	202	REGEL 76	30
REGEL 30	245	REGEL 78	220
REGEL 32	236	REGEL 80	189
REGEL 34	212	REGEL 82	136
REGEL 36	66	REGEL 84	233
REGEL 38	56	REGEL 86	140
REGEL 40	252	REGEL 88	195
REGEL 42	228	REGEL 90	153
REGEL 44	49	REGEL 92	246
REGEL 46	33		
REGEL 48	236	READY.	

Verbeterde herplaatsing van tempo typen C-16 II uit Commodore Info jr. 4 - no. 1 pag. 52.

TEMPO TYPEN C-16 II

Door Frank en Erik van de Pol kregen we weer een andere versie van Tempo typen voor de C-16 toegestuurd. Zij laten ons zien wat er al niet mogelijk is met zo'n beperkt geheugen.
De bedoeling van dit spel is om de woorden die op het scherm verschijnen na te typen voor dat ze door de laserstraal vernietigt worden. Het spel wordt steeds moeilijker naarmate men verder komt. Het voordeel van dit spel is dat hoe meer je dit speelt, hoe sneller je onze volgende listings intypt.

```
1 rem tempo typen / c-16
2 rem
3 rem door erik van de pol
4 rem
5 rem (c) 5-8-1986  by powerso
  ft
6 rem
7 rem ************************
  ********
8 trap9:goto11
9 iferr=30thenresume
10 end
11 gosub152
12 printchr$(8)chr$(14):lvel=10
   0:hi=0:sc=0:t=0:ba=10:hc=5:c
   r=35:goto16
13 color1,8:color4,1:color0,1
14 ifsc>hithenhi=sc
15 sc=0:hc=5:gosub175
16 vol6
17 color1,8:color4,1:color0,1
18 a=rnd(-ti):ifpeek(15620)<>0then
   gosub152
19 hc=hc-1
20 poke65286,0:scnclr:gosub72
21 char1,0,5,"#################
   ########################"
22 char1,0,10,"$$$$$$$$$$$$$$$$
   $$$$$$$$$$$$$$$$$$$$$$$$"
23 ifhc=0andba<20thenlvel=lvel-
   10:ba=ba+2:hc=5:sound1,910,4
   5:sound2,911,45
24 fort=1tocr:char1,t,11,"{GROEN}
   L{GEEL}":next:poke65286,27:if
   mz=0thengosub85
25 char1,20,12,"hoogste score:"
   +str$(hi):mz=1
26 char1,ba,5,"A":char1,ba,10,"
   C"
27 forq=6to9:char1,ba,q,"B":next
28 fort=sctos:char1,1,12,"score
   :"+str$(t):next:forlg=0to10:
   gettb$:next
29 sc=s
30 a=120:restore:p=int(rnd(1)*a
   )+1:forb=1top:reada$:next
31 le=len(a$)
32 forq=1tole:x=37-q:char1,x,7,
   left$(a$,q):next
33 dountilx=1
34 ifx<bathencr=cr-1:exit
35 getz$
36 ifpeek(15620)<>0thengosub152
37 ifz$=left$(a$,1)andlen(a$)>0
   thena$=right$(a$,len(a$)-1):
   s=s+x
38 iflen(a$)=0thensound1,800,20
   :exit
39 char1,x,7,a$+"{2xspatie}":if
   x>1thenx=x-1
40 fort=1tolvel:next
41 loop:iflen(a$)=0then47
42 k=len(a$):fort=1tok:sound3,9
   00+(t*5),10:iflen(a$)>1thena
   $=right$(a$,len(a$)-1)
43 char1,ba-2,6,"MN":char1,ba-2
   ,7,"OP":char1,ba-2,8,"QR"
44 forw=1to50:next:char1,1+x,7,
   a$+"{2xspatie}":cr=cr-1:ifcr
   =>0thenchar1,cr,11,"{3xspatie}"
45 if cr=<1thent=k:next:goto13
46 next
47 goto18
48 data"beestje","handleiding"
49 data"aalscholver","aanfluiti
   ng","aardappel","achtergrond"
   ,"advertentie"
50 data"afkeuren","agressief","
   alpinisme","bakeliet","basil
   iek","bibliotheek"
51 data"beestenboel","benzine",
   "centerboor","choreograaf","
   collectie","comite"
52 data"cursief","dansleraar","
   delfstof","diapositief","din
   osaurus","dossier"
53 data"emulsie","endeldarm","e
   xternaat","exemplaar","ezels
   veulen"
54 data"failiet","filter","fund
   ement","frontlinie","fuut"
55 data"gassluis","gelei","gelu
   iddicht","gemeentehuis","geo
   logie"
56 data"hanekam","hectoliter","
   helium","herrie","huismus"
57 data"idealist","ijsvrij","in
   fluentie","inrit","inversie"
58 data"jachthaven","jichtig","
   jongleren","jurist","juwelie
   r"
59 data"korstje","kaneel","kapi
   taal","kwadraat","klapsigaar"
60 data"lesvliegtuig","levendig"
   ,"lichtbak","lippenstift","l
   ucht"
61 data"mammoet","meander","mic
   robe","methode","musket"
62 data"nimrod","nitraat","nood
   rem","nylon","noteboom"
63 data"octopus","officier","ol
   ieslager","omkopen","ooievaa
   r"
64 data"parachute","pianoles","
   pleonasme","polder","pyromaa
   n"
65 data"raket","radio","rijtuig"
```

```
     ,"roomsoes","royaal"
 66 data"salpeter","schaakspel",
     "scalpel","secundair","slapl
     ant"
 67 data"tautologie","teakhout",
     "telex","titaan","traangas"
 68 data"uitrekKen","urine","uni
     versum","uitzenden","uitkijk"
 69 data"valhelm","vogel","verf"
     ,"verhuur","vouwfiets"
 70 data"waakvlam","wapen","wast
     afel","walm","wedstrijd"
 71 data"zaagsel","zeekoe","zege
     l","zeil","zeskant","eod"
 72 print"{CLR-HOME}":color1,15,
     5
 73 char1,5,0,"IIIII IIIII {RVS-aan}
     _{RVS-uit} {RVS-aan}) {RVS-uit}
     IIIIE DIIIE"
 74 char1,5,1,"{2xspatie}{RVS-aan}
     {RVS-uit}{3xspatie}{RVS-aan}
     {RVS-uit}{5xspatie}{RVS-aan}
     {2xspatie}K{2xspatie}{RVS-uit}
     I{3xspatie}J I{RVS-aan}D{RVS-ui
     t} {RVS-aan}E{RVS-uit}I"
 75 char1,5,2,"{2xspatie}{RVS-aan}
     {RVS-uit}{3xspatie}{RVS-aan}
     {3xspatie}{RVS-uit}{3xspatie}
     {RVS-aan} {RVS-uit}_{RVS-aan}
     {RVS-uit}){RVS-aan} {RVS-uit}
     IIIIG {RVS-aan} {RVS-uit}{3xspa
     tie}{RVS-aan} {RVS-uit}"
 76 char1,5,3,"{2xspatie}{RVS-aan}
     {RVS-uit}{3xspatie}{RVS-aan}
     {RVS-uit}{5xspatie}{RVS-aan}
     {RVS-uit} K {RVS-aan} {RVS-uit}
     {RVS-aan} {RVS-uit}{5xspatie}
     {RVS-aan} F{RVS-uit} {RVS-aan}
     G {RVS-uit}"
 77 char1,5,4,"{2xspatie}{RVS-aan}
     {RVS-uit}{3xspatie}{RVS-aan}
     {5xspatie}{RVS-uit} {RVS-aan}
     {RVS-uit}{3xspatie}{RVS-aan}
     {RVS-uit} {RVS-aan} {RVS-uit}
     {5xspatie}FIIIG"
 78 char1,5,14,"{RVS-aan}{5xspatie}
     {RVS-uit} {RVS-aan} _{RVS-uit}
     {RVS-aan}) {RVS-uit} IIIIE
     {RVS-aan}{5xspatie}{RVS-uit}
     {RVS-aan} _{RVS-uit}{2xspatie}
     {RVS-aan} {RVS-uit}"
 79 char1,5,15,"{2xspatie}{RVS-aan}
     {RVS-uit}{3xspatie}_{RVS-aan}
     K {RVS-uit}) {RVS-aan} {RVS-uit
     }{3xspatie}J {RVS-aan} {RVS-uit}
     {5xspatie}{RVS-aan}{2xspatie}
     _{RVS-uit} {RVS-aan} {RVS-uit}"
 80 char1,5,16,"{2xspatie}{RVS-aan}
     {RVS-uit}{4xspatie}_{RVS-aan}
     {RVS-uit}){2xspatie}IIIIG {RVS-
     aan}{3xspatie}{RVS-uit}{3xspatie
     }{RVS-aan} {RVS-uit}_{RVS-aan}
     _ {RVS-uit}"
 81 char1,5,17,"{2xspatie}{RVS-aan}
     {RVS-uit}{5xspatie}{RVS-aan}
     {RVS-uit}{3xspatie}{RVS-aan}
     {RVS-uit}{5xspatie}{RVS-aan}
     {RVS-uit}{5xspatie}{RVS-aan}
     {RVS-uit} _{RVS-aan}{2xspatie}
     {RVS-uit}"
 82 char1,5,18,"{2xspatie}{RVS-aan}
     {RVS-uit}{5xspatie}{RVS-aan}
     {RVS-uit}{3xspatie}{RVS-aan}
     {RVS-uit}{5xspatie}{RVS-aan}
     {5xspatie}{RVS-uit} {RVS-aan}
     {RVS-uit}{2xspatie}_{RVS-aan}
     {RVS-uit}"
 83 char1,13,20,"{grijs 3}by pow
     er{grijs 1}H{grijs 3}soft"
 84 print"{GEEL}":return
 85 te=8:vol6
 86 sound1,929,te:sound2,739,te*
     2
 87 sound1,881,te/2
 88 sound1,897,te/2
 89 sound1,911,te/2:sound2,770,t
     e
 90 sound1,917,te/2
 91 sound1,929,te:sound2,798,te*
     3
 92 sound1,881,te:sound1,1020,1
 93 sound1,881,te
 94 sound1,939,te:sound2,810,te*
     3
 95 sound1,917,te/2
 96 sound1,929,te/2
 97 sound1,939,te/2
 98 sound1,948,te/2
 99 sound1,953,te:sound2,798,te*
     3
100 sound1,881,te:sound1,1020,1
101 sound1,881,te
102 sound1,917,te:sound2,770,te*
     3
103 sound1,929,te/2
104 sound1,917,te/2
105 sound1,911,te/2
106 sound1,897,te/2
107 sound1,911,te:sound2,739,te*
     3
108 sound1,917,te/2
109 sound1,911,te/2
110 sound1,897,te/2
111 sound1,881,te/2
112 sound1,873,te:sound2,834,te
113 sound1,881,te/2:sound2,798,t
     e
114 sound1,897,te/2
115 sound1,911,te/2:sound2,739,t
     e
116 sound1,881,te/2
117 sound1,897,te*3:sound2,834,t
     e:sound2,643,te/2:sound2,810
     ,te/2:sound2,798,te/2
118 sound2,770,te/2
119 sound1,929,te:sound2,798,te*
     2
120 sound1,881,te/2
121 sound1,897,te/2
122 sound1,911,te/2:sound2,770,t
     e
123 sound1,917,te/2
```

```
124 sound1,929,te:sound2,739,te
125 sound1,881,te:sound2,798,te:
    sound1,1020,1
126 sound1,881,te:sound2,739,te
127 sound1,939,te:sound2,810,te*
    3
128 sound1,917,te/2
129 sound1,929,te/2
130 sound1,939,te/2
131 sound1,948,te/2
132 sound1,953,te:sound2,798,te
133 sound1,881,te:sound2,810,te/
    2-1:sound2,798,te/2
134 sound1,881,te:sound2,770,te/
    2:sound2,739,te/2
135 sound1,917,te:sound2,770,te*
    2
136 sound1,929,te/2
137 sound1,917,te/2
138 sound1,911,te/2:sound2,722,t
    e
139 sound1,897,te/2
140 sound1,911,te:sound2,739,te*
    2
141 sound1,917,te/2
142 sound1,911,te/2
143 sound1,897,te/2:sound2,798,t
    e
144 sound1,881,te/2
145 sound1,897,te:sound2,810,te
146 sound1,911,te/2:sound2,834,t
    e
147 sound1,897,te/2
148 sound1,881,te/2:sound2,634,t
    e
149 sound1,873,te/2
150 sound1,881,te*3:sound2,739,t
    e*2:sound2,453,te
151 return
152 restore
153 poke65286,0
154 reada$:ifa$<>"eod"then154
155 a=1536:do:readc:ifc=555thenexit

156 pokea,c:a=a+1:loop:sys1536
157 data 162,0,189,0,208,157,0,6
    0,189,255,208,157,255,60,189
    ,254,209,157,254,61
158 data 189,253,210,157,253,62,
    232,224,0,208,227,173,18,255
    ,41,251,141,18,255
159 data 173,19,255,41,3,9,60,14
    1,19,255,169,0,133,55,169,60
    ,133,56,96,555
160 fort=65to82:fora=0to7:readc:
    poke15360+a+(8*t),c:next:next

161 fort=48to57:fora=0to7:readc:
    poke15360+a+(8*t),c:next:next
    :poke65286,27
162 return
163 data250,250,116,56,16,16,56,
    16 ,8,16,32,32,32,16,8,8 ,16
    ,56,16,16,56,116,250,250
164 data 7,31,63,127,127,255,255
    ,255 ,224,248,252,254,254,25
    5,255,255 ,255,255,255
165 data127,127,63,31,7 ,255,255
    ,255,254,254,252,248,224,6,1
    5,15,15,240,240,240,96
166 data255,255,255,255,255,255,
    255,255, 255,127,63,63,63,63
    ,127,255
167 data 255,126,60,24,0,0,0,0 ,
    246,238,222,198,246,238,222,
    0
168 data18,36,36,36,36,36,36,36
    ,0,0,64,128,134,144,145,146
169 data 36,36,36,36,36,36,36,36
    ,146,146,146,146,146,146,14
    6,146
170 data36,36,36,36,36,36,36,18
    ,146,145,144,134,128,64,0,0
171 data 126,66,66,66,66,66,126,
    0 ,2,2,2,2,2,2,2,0 ,126,2,2,
    126,64,64,126,0
172 data 126,2,2,126,2,2,126,0
173 data 66,66,66,126,2,2,2,0 ,1
    26,64,64,126,2,2,126,0 ,126,
    64,64,126,66,66,126,0
174 data 126,2,2,2,2,2,2,0 ,126,
    66,66,126,66,66,126,0, 126,6
    6,66,126,2,2,126,0
175 color0,3,3:color4,3,3
176 print"{CLR-HOME}{WIT}":Key1,
    "&":Key2,"\"
177 print"{5xneer}{RVS-aan}f-1{RVS-u
    it}{3xspatie}start"
178 print"{2xneer}{RVS-aan}f-2{RVS-u
    it}{3xspatie}laatste niveau"

179 char1,0,5,"$$$"
180 char1,0,7,"###"
181 char1,0,8,"$$$"
182 char1,0,10,"###"
183 getKeya$
184 ifa$="&"then187
185 ifa$="\"then188
186 goto183
187 ba=10:cr=35:lvel=100
188 t=0:s=0:cr=35:return
```

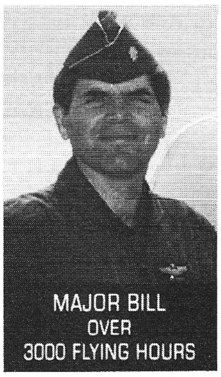
*** EINDE LISTING ***

Neem een ervaren jachtvlieger, een meesterprogrammeur voor simulatiespelen en een ondernemend softwarehuis en je hebt een dijk van een succesformule. Dit geldt althans voor de nieuwe generatie vluchtsimulatoren van Micro Prose. Dankzij de ruime praktijkervaring van de ex-jachtvlieger "Wild" Bill Stealy en de talenten van het simulatiegenie Sid Meier werd de wereld van gierende jets, bliksemsnelle (gevechts-)-acties en adembenemende stunts werkelijkheid op de C-64 en C-128.

Micro Prose's vluchtsimulatoren

De digitale vogels van Bill Stealy en Sid Meier





Het Amerikaanse softwarehuis Micro Prose heeft een gedegen reputatie verworven op het gebied van levensechte simulatiespelen op de Commodore. Wargames zoals Silent Service, Destroyer Escort en Nato Commander zijn tot in de details natuurgetrouw uitgewerkt en historisch korrekt. Behalve "ter land en ter zee" houden de programmeurs en ideeënmakers van Micro Prose zich echter ook bezig met "in de lucht". Een indrukwekkende lijst van real time vluchtsimulatoren veroverde de markt stormenderhand. Kwaliteitssoftware die de wereld van de jacht-, post- en verkeersvliegers met een zinnenprikkelende realiteit in de huiskamer brengt.

Bill Stealy stichtte samen met zijn vriend Sid Meier ongeveer vier jaar geleden het softwarehuis Micro Prose. "Wild" Bill is een ex-jachtvlieger die zich, na het verlaten van de actieve dienst, heeft toegelegd op het ontwerpen van computersimulaties. Daarnaast adviseert hij nog het Pentagon, de Amerikaanse marine en luchtmacht.
Voor het ontwerpen van een C-64/128-simulatie gaan Bill en zijn chef-programmeur nooit over één nacht ijs. Voor het ontwerp van **Silent Service** werden bijvoorbeeld twee onderzeebootexperts geconsulteerd. Volgens Wild Bill is de realiteit de sleutel tot succes. Op de vooravond van de oprichting van Micro Prose speelden hij en Sid Meier in Las Vegas het Atari WOI-arcadespel **Red Baron**. Sid merkte op dat hij een veel realistischer versie in een week kon schrijven. Bill zei daarop:"Als jou dat lukt verkoop ik dat spel aan elke officiersclub in de wereld".

Na twee maanden hard werken lag er de realistische WOII-vluchtsimulator **Hellcat Ace** in de winkel en Micro Prose was geboren.
Sinds de begindagen houdt Micro Prose zich voornamelijk met wargames en vluchtsimulatoren bezig. De oorlogssimulaties zijn zeer succesvol, maar vlak ook de flight simulators niet uit. Daar zijn er inmiddels honderduizenden van verkocht. Voor ieder die naar het gieren der jets, de stank van kerosine en adembenemende vluchten snakt geven wij hier een korte bloemlezing van Micro Prose's vluchtsimulatoren.

F-15 Strike Eagle

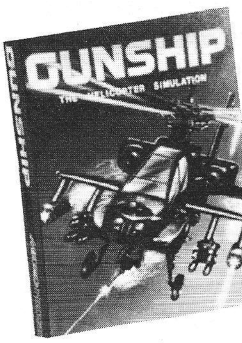
Bill's favoriete flight simulator is **F-15 Strike Eagle.** De speler belandt na het laden van het game in de cockpit van een van de meest geavanceerde gevechtsmachines ter wereld, de F-15. Een F-15 is een ontzagwekkend wa-

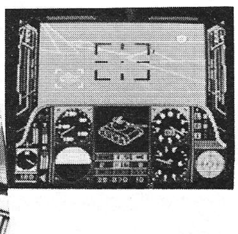
pen, maar al die gevechtskracht vereist ook een vaardige hand, inzicht en veel vliegervaring.
Vervelen zal de speler zich niet gauw. Om de kist zonder crashes van de grond te krijgen zal de leerling-piloot eerst iets van de aerodynamiek, snelheidscontrole en het brandstofverbuik van de jager moeten leren. Daarna mogen de zeer realistische gevechtsacties uitgevoerd worden. Her wargame is verrassend snel. De handelingen lopen ongeveer viermaal zo vlot als bij een gewoon arcade-game en de speler heeft zijn of haar handen meer dan vol aan de actie.
Ondanks het feit dat F-15 Strike Eagle een gevechtssimulator tegen (niet zo) fictieve doelen is biedt het pakket tevens een flinke dosis vliegtraining. Het instrumentenpaneel is uiteraard eenvoudiger dan in de echte jager, maar biedt voldoende vluchtinformatie voor de beginnende piloot. Zodra de speler weet wat de gevechtsmachine kan is er keuze uit diverse oorlogshandelingen. Bijvoorbeeld luchtgevechten, het aanvallen van doelen in Zuid-Oost Azië of het Midden-Oosten. Er zit zelfs een waar gebeurd incident voor de kust van Libië bij.Op de monitor wordt de vlieger goed geïnformeerd over de eigen bewapening, vluchtcondities, het doelgebied en de positie van de vijandelijk machines. Waarschuwingsboodschappen brengen precaire situaties onder de aandacht van de Commodore-piloot.
De graphics zien er, ondanks het feit dat dit wargame al weer twee jaar meedraait, nog redelijk uit. Mogelijk kunnen zij bij de opvolger wat minder ruw. Het geluid is functioneel. Gewoon een van de betere combat flight simulators.

Gunship

De dagen dat de cavalerie de vijand met de blanke sabel te lijf ging zijn voorbij. Het moderne leger gebruikt zwaar bewapende helicopters, zogenaamde gunships, ter ondersteuning van de infanterie. Micro Prose's **Gunship** is een perfecte 3D-simulatie van zo'n gevechtshelicopter.
AH-64 Apache Gunship is duidelijk een meer up-to-date flight simulator dan de F-15 Strike Eagle. De graphics zijn gedetailleerder, veelzijdiger (het instrumentenpaneel is een waar meesterwerkje op de Commodore) en 3D. Met name het 3D-aspect maakt het vliegen zeer realistisch. De piloot kan nu "echt" met de joystick tussen gebouwen, bomen, heuvels, e.d. doorvliegen. Alle scenery schiet met een real time vliegsnelheid voorbij en een foutje is zo gemaakt. Leer daarom eerst vliegen alvorens u een missie gaat vervullen!
Het Apache Gunship is nog dodelijker dan de gelijknamige Indianenstam. Met behulp van het Integrated Helmet Targeting System (IHMS) kan de vlieger de 30mm snelvuurkanonnen en de Hellfire air-to-air raketten nauwkeurig richten. Het opsporen van de vijand wordt vergemakkelijkt door het gebruik van TV-gunsight en laser-afstandmeting.
De vijand is echter ook niet voor de poes. Guerilla's met machinegeweren en Russische luchtdoelraketten lusten het onvoorzichtige gunship rauw. En de gevoelige detectoren, zware bepantsering, stoorzenders, en lokvogels garanderen geenszins het eeuwige leven. De bekwaamheid van de speler bij de diverse vlucht- en aanvalsmanoeuvres moet de doorslag geven.Een werkelijk unieke helicoptervluchtsimulator voor de Commodore.



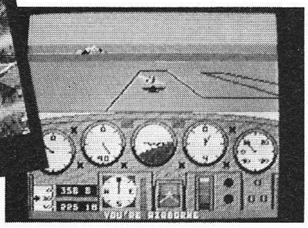




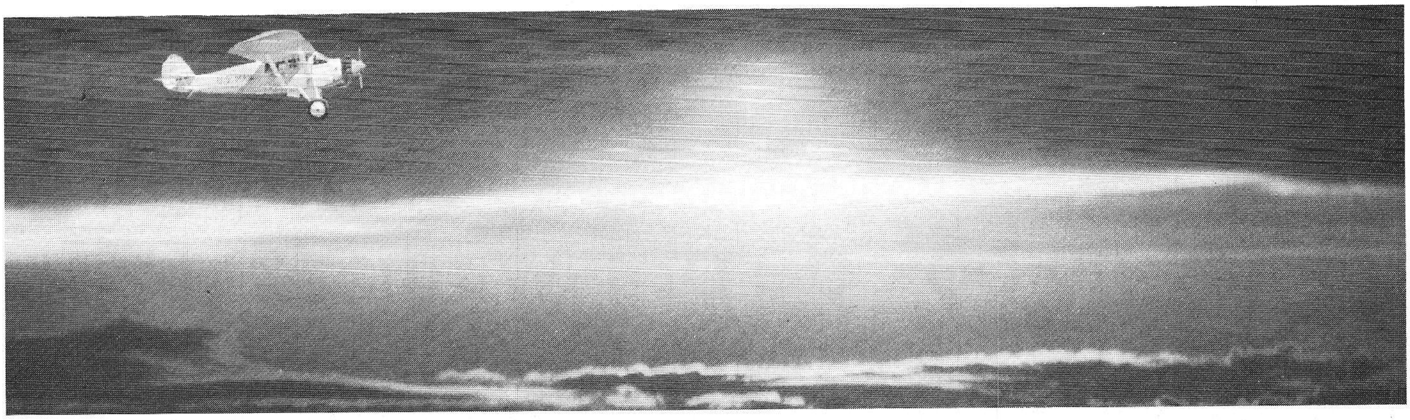
Solo Flight

Het programma **Solo Flight** is waarschijnlijk de beroemdste flight simulator voor de Commodore. In tegenstelling tot de beide vorige pakketten gaat het niet om gevechtsvluchten, maar om commerciële postvluchten onder wisselende zicht- en weersomstandigheden. Ook kunnen een aantal vormen van pech realistisch gesimuleerd worden.



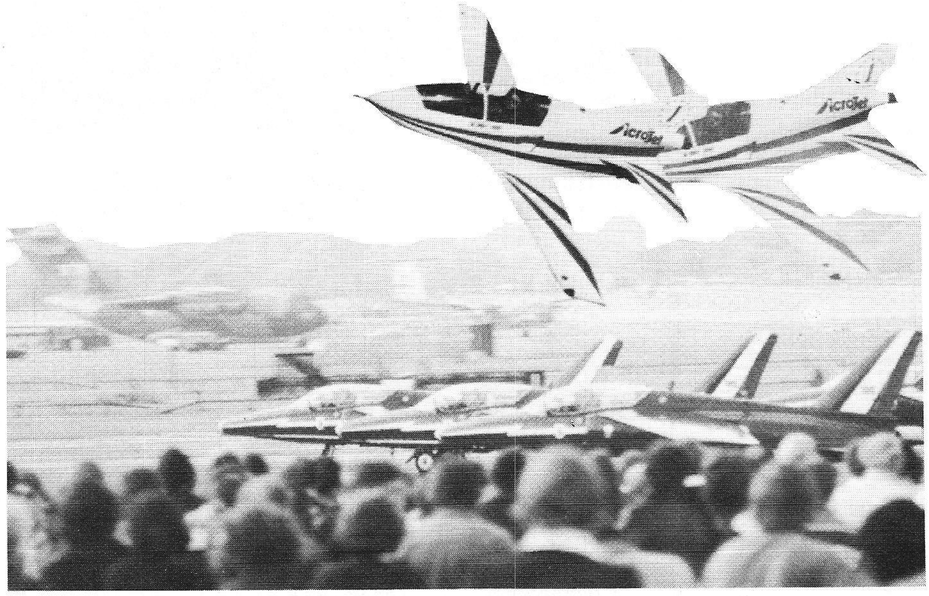


Solo Flight is inmiddels aan versie 2 toe. De graphics werden verbeterd (het cockpit interieur is gedetailleerder en de scenery 3D en wat rustiger geworden), de navigatiekaarten uitgebreid en het monotone motorgebrom aangevuld met spraaksynthese. De vlieger kan nu echt in het Engels met de verkeerstoren communiceren.Solo Flight richt zich vooral op de vliegpraktijk en -oefening, hoewel er ook gewoon voor de lol gevlogen kan worden. De leerling piloot practiseert het opstijgen/landen, het vliegen bij dag/nacht, het binnenkomen op de instrumenten, het navigeren en het handelen in noodsituaties. Er is keuze uit 21 vliegvelden, dag/nachtvluchten, diverse weersomstandigheden en een aantal noodgevallen om het geleerde in praktijk te brengen. Voor een zo groot mogelijke realiteit lieten Meier en Stealy het pakket door zo'n dertig ervaren vliegers testen.
Soloflight is geen toonbeeld van fraaie



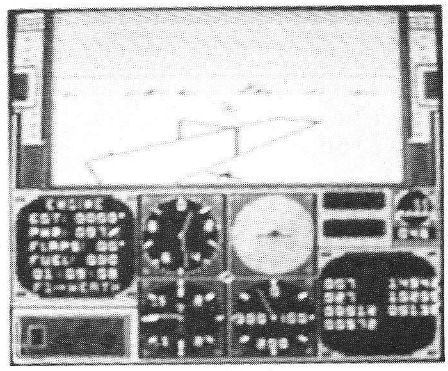
graphics en flitsende actie, maar een solide lespakket. Aanbevolen voor hen die naast de lol ook nog veel van het vliegen zelf willen leren. Gesp uw parachute om een start de motoren. Het avontuur van het zwerk lokt.

Solo Flight draait op elke van een joystick voorziene C-64 en C-128.



Acrojet

Voor hen die al dat geknok of mooiweer-vliegen maar gewoontjes vinden biedt Micro Prose het nodige stuntwerk in **Acrojet.** Een ware decathlon in de sportaviatiek.In Acrojet, de BD5-J éénpersoonsjet, mag de ondernemende Commodore-stunter een tiental gewaagde manoeuvres volbrengen. Wat dacht u van nauwkeurige rollen, loopings, Immelmannen uit de hoge



school van de luchtacrobatiek en de doodtartende Inverted Ribbon Cut? Een fluitje van een cent? Dat valt in de praktijk zwaar tegen. Gelukkig zijn de crashes op de Commodore niet echt. Toch vormt elke crash een smet op het blazoen van de digitale stuntvlieger en stimuleert tot een betere vliegtraining. Ook Acrojet biedt weer gedetailleerde 3D-cockpit graphics (de scenery blijft tegenvallen), real time vliegsimulatie en slechts functioneel geluid. Een aardige toevoeging is het wedstrijdelement in de vorm van vier concurrerende spelers.

Bliksemsnelle joystickacties, vliegge-voel en -ervaring beslissen het pleit voor de winnaar. Een vergissinkje betekent al een jammerlijke crash en GAME OVER.

Bent u uitgekeken op de gewone vluchtsimulatoren of zoekt u een vervolg op Solo Flight, gesp dan het valschermharnas om, schuif de blokken voor de wielen weg, start de jet en doe een schietgebedje. Een adembenemende vliegervaring wacht.

Kennedy Approach

Dit maal geen echte vluchtsimulator, maar een realistische benadering van het boeiende verkeersleiderswerk compleet met rampenscenario. Kennedy Airport is een van de drukste vliegvelden ter wereld en de spanningen kunnen zenuwslopend hoog oplopen. Proef de sensatie van de spanning bij het leiden van druk en gecompliceerd luchtverkeer of een dreigende ramp. Ontwerper Andy Hollis en Sid Meier maakten een prijswinner van deze verkeerstorensimulatie.

Als verkeersleider komt de speler ogen en oren te kort. Twintig verschillende vliegtuigen willen tegelijk en liefst op dezelfde baan, landen. Druk radioverkeer (Engelse spraaksynthese) moet beantwoord en de radarmeldingen nauwlettend geïntepreteerd

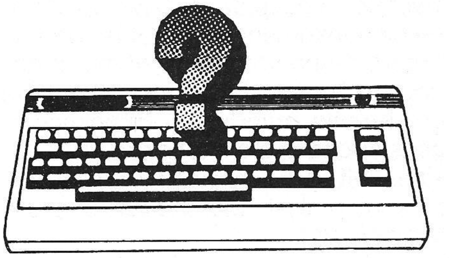




worden. De veiligheid van vele honderden passagiers en bemanningsleden ligt in de zwetende handen van de Commodore-verkeersleider/ster.
Ter verhoging van de spanning en de realiteit krijgt de verkeerstoren nog te maken met een dreigende crash. Net als in een klassieke rampenfilm loopt de temperatuur in het huiskamervluchtleidingscentrum danig op. Haalt de binnenkomende jet met brand aan boord, brandstof te kort of motorstoring de landingsbaan nog veilig? De crash-units, brandweer- en ziekenwagens spoeden zich naar hun rampposities. Dit alles in functionele 3D-graphics en omlijst door realistische geluidseffecten.
Een educatieve en vermakelijke simulatie van het verkeersleiderswerk. ●

De vluchtsimulatoren van Micro Prose hebben door hun realiteitszin het digitale luchtruim veroverd. Menige Commodore-computeraar maakte door MP's simulatoren kennis met het avontuur van het luchtruim en stak daar nog het een en ander van op ook.
Vliegen op de huiscomputer betekent met de nieuwe generatie flightsimulators meer dan het spelen van een simpel arcadegame. Het echte leunstoelvliegen op de C-64 en C-128 is nu binnen ieders bereik gekomen.

Vragen van Gebruikers.



Ook nu weer gaat Jan Bodzinga in op de problemen van onze lezers. Zijn rubriek blijkt zeer populair, maar laat u dat niet weerhouden uw vragen of opmerkingen in te sturen. Ook tips en handigheidjes zijn welkom..

GET met cursor

P.Pukkel uit Grootegast vraagt of het mogelijk is, om in Basic de C-64 te programmeren voor input met hulp van de GET opdracht, zodat er steeds één karakter van het toetsenbord wordt gevraagd en kan worden getest, terwijl dan wel de cursor op het scherm blijft knipperen.

De GET opdracht vraagt in principe zonder cursor een karakter. Iedere toets die wordt ingedrukt door de gebruiker wordt door het programma gelezen. Ook als er geen toets wordt ingedrukt zal het programma na een GET A$ verder gaan, met in de variabele A$ een nulstring (""). Wil je dus wachten op een input bij GET-opdrachten, dan moet het programma er zo uitzien:

```
100 PRINT "KORREKT J/N "
110 GET A$: IF A$ = "" THEN 110
120 IF A$ AND 127 ="j" THEN PRINT
    "JA": GOTO 150
130 IF A$ AND 127 ="n" THEN PRINT
    "NEE": GOTO 150
140 PRINT "FOUT" : GOTO 100
150 rest programma...
```

Dit alles gebeurt echter zonder dat er een cursor op het scherm verschijnt. Om de cursor geforceerd 'aan' te zetten, tijdens het wachten op de invoer van het keyboard, moeten we een adres in de ZERO-PAGE van de C-64 poken. Op adres $CC (204 decimaal) bevindt zich de vlag, die de C-64 vertelt of de cursor aan of uit moet zijn. Als de waarde in 204 nul is, zal de cursor knipperen. Is die waarde ongelijk aan nul, dan is de cursor niet zichtbaar op het scherm. Bovenstaand programma kunnen we nu aanpassen om hetzelfde met knipperende cursor te bewerkstelligen :

```
100 PRINT "KORREKT J/N "
110 GET A$: IF A$ = "" THEN POKE
    204,0: GOTO 110
120 IF A$ AND 127 ="j" THEN PRINT
    "JA": GOTO 150
130 IF A$ AND 127 ="n" THEN PRINT
    "NEE": GOTO 150
140 PRINT "FOUT" : GOTO 100
150 POKE 204,1 : REM CURSOR UIT
```

In regel 110 wordt nu door POKE 204,0 de cursor steeds aangezet, terwijl bij het afwerken van de invoer in regel 150 de cursor weer wordt uitgeschakeld.

2 Computers 1 drive ?

Van diverse kanten komt de vraag of het mogelijk is om zowel de 128 als de 64 tegelijkertijd met dezelfde diskdrive te laten werken. Alex van den Bogaerdt uit Zoetermeer wil daarnaast graag weten of de 1571 ook met de C-64 kan werken.

Om met de laatste vraag te beginnen, het is inderdaad mogelijk om de 1571 diskdrive van Commodore samen met de C-64 te gebruiken. Je hebt er alleen niet erg veel voordelen van, omdat de snelheid van deze superdrive(?) automatisch wordt aangepast aan de langzame C-64. Wel kun je met deze configuratie over een grotere diskcapaciteit beschikken. Om naast een C-128 ook een C-64 te laten werken met dezelfde randdapparaten is in principe even mogelijk als onmogelijk. Omdat er in de randapparaten geen manier aanwezig is om te bepalen van welek computer er opdrachten binnenkomen, kan er altijd maar één computer tegelijk communiceren met dezelfde periferie. Zodra beide computers op **hetzelfde** moment de drive willen gebruiken, blijft er van de diskette-data in ieder geval weinig bruikbaars over. De hardware koppeling van diverse computers en randapparaten aan elkaar is in feite geen probleem, als je maar een gedisciplineerd gebruik maakt van de onderlinge communicatiemogelijkheden.

40-koloms scherm ?

S.van Leeuwen komt met de opmerking, dat er van de 40 regelposities op het scherm in feite maar 39 volledig gebruikt kunnen worden, omdat bij het printen van de laatste positie de cursor automatisch naar de volgende regel verspringt. Dit leidt vooral bij de onderste regel tot problemen, omdat de display dan gaat scrollen.

Om van het hele scherm een efficient gebruikt te kunnen maken, kun je b.v. de laatste positie POKEN in plaats van printen. Wil je het hele scherm volzetten met een bepaald karakter, dan is dit natuurlijk de oplossing. De poke-adressen van het scherm zitten tussen (decimaal) 1024 en 2023. Om de laatste (40e) regelpositie te vinden kun je gebruik maken van de volgende Basic-loop:

```
100 FOR X = (1024+39) TO 2023
    STEP 40: PRINT X (of POKE X,a-
    sciiwaarde karakter): NEXT
```

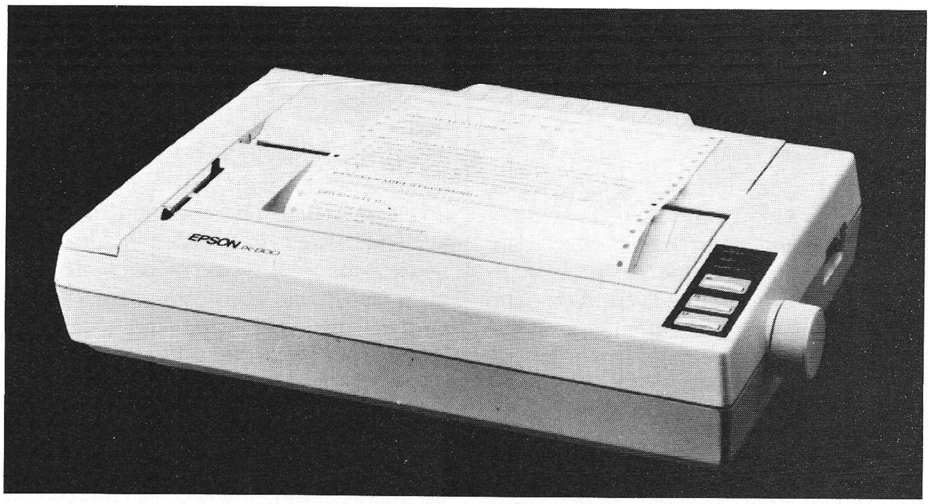
Deze methode is echter niet geschikt voor het printen van teksten.
Maar ook daar is een oplossing voor, en dat is het werken met de puntkomma achter een printopdracht. Bekijk het verschil in uitvoer maar eens tussen de volgende twee printregels.

```
10A$="Dit is een test "
20 FOR X = 1 TO 5 : PRINT A$ : NEXT
30 PRINT
40 FOR I = 1 TO 5 : PRINT A$; : NEXT :
   REM DENK AAN PUNTKOMMA
```

Je zult direkt zien, dat bij de loop van regel 40 geen nieuwe regel wordt gebruikt om A$ te printen. De puntkomma betekent voor de Basic-vertaler zoveel als : Blijf staan op dezelfde plaats, tot er weer geprint moet worden. Daardoor ben je in staat om ook op de onderste regel, tot aan de laatste positie teksten op het scherm te zetten. Sluit daartoe wel iedere PRINT af met een puntkomma.

Als laatste in de reeks van drie artikelen over de meest gekochte hardware uitbreidingen voor de Commodore 64, volgt nu een artikel over Printer Interfaces.

Printer Interfaces



Voor degenen die zijn uitgekeken op de printers die Commodore voor de CBM-64 levert of die dezelfde printer (in de toekomst) ook op een andere computer willen aansluiten, is de aanschaf van een printerinterface een vrijwel onontkoombare zaak.

De interfaces die voor de CBM-64 leverbaar zijn vallen uiteen in software interfaces en hardware interfaces. Over de eerste groep wil ik kort zijn, dit ondanks de scherpe prijsstelling ten opzichte van de hardware interfaces. Deze groep levert nl. de meeste problemen op die uit het volgende voortkomen: als hardware heeft men aan een simpele Centronics kabel van ongeveer ƒ 75,-, die men meestal aan de Userport bevestigt, voldoende. De verdere aansturing geschiedt via de nodige software. Het is nu deze software die de grootste problemen oplevert. Het moet nl. ergens in het geheugen van de CBM-64 ondergebracht worden. Meestal wordt hiervoor het gebied vanaf Hex C000 gebruikt. De programma's die geen gebruik maken van dit geheugengebied zijn meestal de in Basic geschreven programma's. De meeste professionele software maakt echter WEL gebruik van dit gebied. Het resultaat is dat je GEEN LETTER op je printer kan laten verschijnen. Een ander nadeel is dat je niet meer over een seriële poort beschikt als je printer staat aangesloten (dus geen modem). Vandaar dat we onze aandacht beter kunnen richten op de tweede groep van interfaces. Hierbij diende ik een persoonlijke selectie te maken, dit om het artikel binnen bepaalde grenzen te houden.

Mijn voorkeur ging uit naar de volgende 5 interfaces:

- Cardco P/S met zowel een seriële als een parallelle uitgang. Prijs: ƒ 365,- (Funtronics).
- Cardco Card/Print +G. Prijs: ca. $ 89,-
- Wiesemann WW 92000/G. Prijs: DM 248,-
- Zero C-64 parallel interface. Prijs: ƒ 145,- (Zero S.C.).
- Gorlitz VCEI (voor Epson). Prijs: DM 298,-

Alvorens deze vijf apparaten uitvoerig te behandelen wil ik nog even de werking en de overeenkomsten van deze interfaces beschrijven.

Al deze interfaces worden op de seriële bus van de CBM-64 aangesloten. De voeding kan van de volgende 3 plaatsen komen: de cassettepoort, de printer of een externe voeding.

De interface bestaat uit niets anders dan een eenvoudig microprocessor systeem. Die van Cardco bestaan zelfs uit slechts een (Costom made) IC waarop de Processor + RAM + ROM etc. zijn geïntegreerd. Op de meeste van deze interfaces zijn dipswitches gemonteerd die het mogelijk maken om bepaalde functies, zoals bv. device number, in te stellen.

De interfaces beschikken over verschillende operating-modes. De volgende "modes" zijn het belangrijkst.

- Listing Mode: hierbij worden de grafische tekens bij het listen van een programma omgezet in een reeks speciale karakters. Het voordeel hiervan is dat indien men logische karakters gebruikt (bv. het cursor up teken vervangen door 'CRSRUP'), zo'n listing veel leesbaarder wordt.
- Transparent Mode: in deze modus worden alle karakters letterlijk (dus zonder enige conversie) naar de printer gestuurd. Hierdoor is het mogelijk om o.a. graphics op papier te krijgen.
- Text Mode: In deze modus kan men de normale ASCII karakterset op papier krijgen. Hierbij kan men bovendien de speciale printer stuurcodes gebruiken.

De interfaces met grafische mogelijkheden kunnen bovendien de speciale grafische karakters en soms ook reversed karakters printen. Sommige bieden ook de mogelijkheid van een graphics screendump. Het zal duidelijk zijn dat deze grafische mogelijkheden alleen van toepassing zijn op een bepaald type printer (Epson en compatibles).

Door enkele van deze interfaces wordt bovendien de mogelijkheid geboden om de laatst ingestelde operating-mode zogenaamd te "locken". Deze laatste modus kan men alleen wijzigen door de computer uit te zetten. Dit is erg handig omdat je zo het overschrijven door bv. een tekstverwerker kan voorkomen.

Cardco P/S

Dit is een van de weinige interfaces die van de seriële uitgang van de CBM-64 een RS 232 uitgang maken. Er zijn echter enkele beperkingen. Als baudrate kan men alleen tussen 300 en 1200 baud kiezen. Dit zijn echter gebruikelijke snelheden voor seriële printers. Als protocol wordt verder gebruikt:

- No parity.
- 8-bits per word.
- 2 stop bits.

Verder biedt deze interface de volgende voordelen:

- Listingmode. Hierbij worden logische karakters gebruikt.
- Transparentmode.
- Textmode (diverse).
- 'Locken' van de laatst gekozen instelling is mogelijk.

- Device number 4 en 5.
- Voeding via doorgevoerde cassette poort.
- 6 dip-switches voor verschillende instellingen. De dip-switch instellingen hebben een hogere prioriteit dan de instellingen via software.
- Is getest met een hele serie printers waaronder o.a. Epson, Star, C.Itoh, Seikosha, Okidata etc.

Verder dient de uitstekende handleiding vermeld te worden, die in feite een mini-cursus printgebruik bevat.

Card/Print +G

Dit is een soort grafische versie van de Cardco P/S .Nogal wat software van Amerikaanse origine is aan deze interface aangepast (bv. Koala printer). Als grote voordeel van de Card Print +G dient de compatibiliteit met bijna alle printers met een Centronics ingang vermeld te worden.Bijvoorbeeld: Epson MX/RX/FX serie, Star Gemini 10X/15X, Delta 10, C.Itoh 8510, NEC 8023, OKI ML 82 tot en met 94, Mannesmann-Tally Spirit en MT 16.

Het tweede voordeel is dat het een CBM-printer volledig emuleert. De printer stuurcodes van CBM printers worden hierbij zo omgezet dat de ingestelde printer die verstaat. Hierdoor loopt een hele reeks programma's zonder meer (m.n. tekstverwerkers).

Verder gelden als voordelen:
- Voeding via doorgevoerde cassettepoort. - Listingmode. Hierbij worden logische karakters gebruikt. - Print ook grafische karakters. - 8 dip-switches voor de verschillende instellingen. - Prima manual.

Een zwak punt van deze interface is de hardcopy functie. Deze functioneert alleen met Seikosha en compatible printers, al geldt dit natuurlijk niet voor de software die deze interface ondersteunt. Men kan ook makkelijk zelf een programma schrijven om graphics op papier te krijgen.

Wiesemann WW92000/G

Wiesemann biedt verschillende interfaces aan.Verreweg de beste is echter het bovengenoemde produkt. Een sterk punt van de 92000/G is dat het de veelzijdige Commodore VC-1515 printer emuleert. Verder bezit deze interface de volgende eigenschappen: - Voeding via doorgevoerde cassettepoort. - Printer instelling via 6 dip-switches. - "Locken" van de laatst ingestelde mode. - Grafische mogelijkheden op Centronics compatible printers. Op deze printers kan men o.a. een hardcopy krijgen.Reversed karakters zijn bovendien ook nog mogelijk. Graphics in zowel 6-pins als 8-pins mode. - Device number 4 en 5. - Veel tekstverwerkers werken probleemloos in combinatie met deze interface (o.a. Vizawrite, Textomat, Wordcraft en Blitztext) - Uitvoerige handleiding.

Zero Parallel Interface

De interface van Zero S.C. valt meteen op door zijn relatief lage prijs. Tegen een meerprijs van ongeveer *f* 60,- is er ook nog een versie met een ingebouwde buffer van 16 Kb te krijgen.

Deze lagere prijs heeft echter duidelijk invloed gehad op de kwaliteit van de interface. Voorbeeldje: voor het veranderen van het device number van 4 in 5, dient men het kastje te openen met de soldeerbout in de aanslag. Een simpele schakelaar kon er blijkbaar niet meer bij.

De voeding komt van de doorgevoerde cassettepoort. De interface heeft 3 operating modes. Twee van deze modi zijn in feite listingmodes. In beide gevallen worden nl. de ASCII-codes van CHR$ (0) t/m CHR$ (31) (en de codes van CHR$ (128) e.v.) omgezet in een aantal niet erg logisch gekozen karakters. Hierdoor is het in feite onmogelijk om printer-stuurcodes te verzenden zonder dat deze in een idiote reeks karakters worden vertaald.

De derde modus is een transparent mode. De mogelijkheid bestaat om deze modus min of meer te "locken".

Van de compatibiliteit met bestaande tekstverwerkers moet men dus niet al te veel verwachten. Deze zal voornamelijk beperkt blijven tot de normale ASCII tekens (m.a.w. Expanded, NLQ, Boldface etc. kan men wel vergeten).

Vergeleken met de 3 vorige interfaces stelt het Zero-model erg teleur. De nogal wazig overkomende (slecht gecopieerde) handleiding versterkt dit beeld alleen maar.

Gorlitz VCEI

Als laatste interface wil ik de Gorlitz VCEI noemen, die alleen voor inbouw in een printer geschikt is.Dit lijkt misschien niet zo handig ware het niet dat het ontworpen is voor de Epson MX/RX/FX-80 printers. In deze printers hoeft men de kaart er alleen maar in te steken. De voordelen van deze kaart zijn:
- Emuleert een CBM-printer. - Print zowel grafische als reversed karakters. - De hele CBM-karakterset kan op 40 verschillende wijzen geprint worden. - Listingmode. - Graphicsmode. - Textmode. - Hardcopy o.a. mogelijk met Simon's Basic. - Alle Epson stuurcodes blijven werken. - Device number 0-15. - Ingebouwd 2 Kb buffer. - Interface is uit te schakelen zonder de kaart te verwijderen. - Een doorgevoerde seriële bus. ●

P. Sjauw En Wa.

Ocean behoort tot de grootste Engelse Softwarehuizen. Zelf claimt het bedrijf de ongekroonde marktleider te zijn, maar daar over valt natuurlijk te twisten. Qua reklame- en verkoopagressiviteit staat Ocean echter wel degelijk aan de top. Ulco Schuurmans over de methodes en achtergronden van dit geruchtmakende softwarehuis.

Ocean Software

Rambo in Softwareland

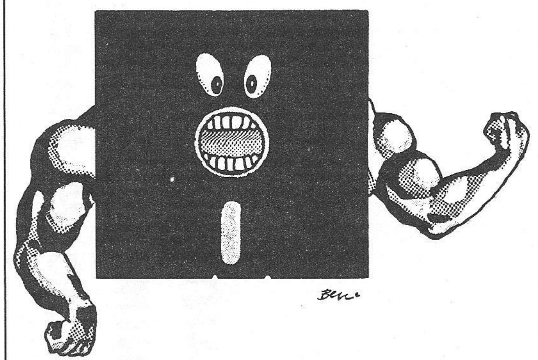
Ocean heeft in Engeland een sterke marktpositie in de videogames veroverd. Niet alleen door de eigen activiteiten, maar ook dankzij de dochteronderneming Imagine, een flinke vinger in de pap bij Konami en de associatie met US Gold van Geoff Brown. Volgens Oceaan zelf goed voor zo'n 30% van de spelletjesmarkt.

Een goede marktpositie is, naast de genoemde samenwerking, ook afhankelijk van het geboden produkt. En dat mag er zijn, gezien de successen van spelen als Rambo, Miami Vice en Cobra. Meestal dus videogames gebaseerd op bekende speelfilms en televisieseries. Niet het allercreatiefste genre, maar wel kassuccessen en daar draait het om, vindt Ocean.

Rambo
De huidige presidentdirekteur, David Ward, noemt zich zelf de Rambo van de Britse spelletjeswereld. Dat Rambo slaat overigens meer op zijn verkooptechnieken dan op Ward's figuur. David Ward is gewoon een van de taaiste en meest meedogenloze softwarebazen uit het gameswereldje.

Het begin
In juli 1983 startte Ocean Software door gewoon simpel een advertentie voor Spectrum Software in de krant te zetten. De respons was enorm en Ward slaagde er in om programmeurs zonder enig zakelijk inzicht te organiseren tot een geoliede verkoopmij. Na de Spectrum volgden Commodore (C-64, C-128, C-16, Plus 4), de Amstrad en schoorvoetend ook MSX. Stoelend op de eerder opgedane ervaringen met boetieks grammofoonplaten en ook nachtclubs wist Ward de vruchtbare akker van de videogames binnen korte tijd te ontginnen. Voor succes in software hoef je dus blijkbaar geen verstand van software te hebben. Gezond zakelijk inzicht is voldoende. De huisprogrammeurs doen de rest.

Toch creatief?
David Ward ziet de zaken zelf enigzins anders. Volgens hem was tot voor tien jaar geleden de menselijke creativiteit het terrein van de kunstenaars, schrijvers en musici. De populariteit van de huiscomputers bracht daar verandering in. Iedereen kan zich nu op de eigen computer uitleven en zich in een nieuwe (game)wereld wanen of zelf een digitaal produkt ontwerpen. Bovendien combineert Ocean, zoals vele andere softwarehuizen, de talenten van meerdere programmeurs tot een geslaagd spel. In deze collectieve creativiteit, de één doet de graphics, de ander het verhaal en weer een ander de muziek of geluiden, kan iets moois opbloeien. Vooral daar Ocean de programmeurs alle hard- en softwarefaciliteiten biedt die zij maar wensen. Een éénling komt meestal lang niet zo ver.



8-bitter gaat nog jaren mee!
Ondanks de pessimistische voorspellingen dat de PCs, ATs, Amiga's en soortgelijke machines de Commodore 64/C128 uit de huiskamer zullen verdringen, voorziet Ocean nog een zonnige toekomst voor de 8-bitter.

Het grote aantal verkochte machines en de lage softwareprijs houden de marktvraag voor de huiscomputer nog steeds in stand. Met name voor de krap gebugetteerde hobbyist, de spelletjesliefhebber en het sterk bezuinigende onderwijs.
Zelf is Ocean het stadium van de huiscomputer al twee jaar ontgroeid. Het heeft geen zin om het wiel

opnieuw uit te vinden. Krachtiger Pc's, tal van beschikbare handige subroutines en utilities maken het ontwerpen van goede spelsoftware aanzienlijk eenvoudiger dan het geploeter in machinetaal op een 64-tje. Na het uitwerken van het basisconcept volgt aanpassing aan de doelmachine en diens eigen specifieke processor.
Binnenkort krijgt Ocean zelfs de beschikking over een uitgebreid VAX-rekenmonster voor de ontwikkeling van nog meer geavanceerde games.

Programmeurs van overal

Ocean wacht niet tot programmeurs bescheiden aan de firmadeur kloppen. Dit Engelse softwarehuis sleept talent via advertenties aan de spreekwoordelijke haren naar binnen. Concurrerende softwarefirma's beklaagden zich in het verleden zelfs over "wegkooptechnieken".
Volgens Ocean komen de programmeurs echter zelf naar het bedrijf met half afgewerkte programma's om deze dan technisch te kunnen vervolmaken. Ocean biedt de spelontwerpers, indien economisch zinvol, de nodige en verzorgt de vertaling. Met deze aanpak lokt het softwarehuis programmeurs vanuit geheel Europa. Ze komen zelfs uit het Oostblok. En daar schuilt veel onvermoed talent onder. Inmiddels beschikt Ocean over ruim 20 vaste programmeurs, een aantal artiesten, diverse musici en enkele animators.

De succesformule

Het succes van Ocean steunt op meerdere marketingpilaren, waarvan sommige nu niet erg creatief genoemd kunnen worden. De eerste successen werden geboekt middels het opnieuw en voor andere machines uitbrengen van oude toppers. Een gerenommeerde techniek uit de platenwereld. Daarna volgde de succesvolle serie sportgames van Daley Thompson. En de laatste tijd zette Ocean veel bekende TV-series en speelfilms om tot computerspelen.
De resultaten kunt u in de vorm van games als **KnightRider**, **Rambo** en **Cobra** in de computershop bewonderen. Een andere belangrijke marketingpilaar is de samenwerking met andere softwarehuizen zoals bijvoorbeeld het fameuze US Gold. Ook worden tal van spelen in licentie verkocht aan Europese softwareleveranciers, die de games weer voor de eigen thuismarkt omzetten.

Of de succesformule van Ocean bestand blijft tegen de tanende belangstelling voor computergames zal de tijd leren. Ook Ocean heeft inmiddels de wispelturigheid van de markt aan den lijve ondervonden. Knight Rider is in Engeland geflopt en doet ook in ons land weinig. Wellicht zijn de dagen van de gameboeren en software-Rambo's geteld, maar de markt kan ook weer snel opleven. •



LUC SALA'S DATAKOLOM

De allernieuwste trend bij video-cassetterecorders is het inbouwen van een VCR in de kast van de huiskamer-TV. Alles gemakkelijk bij elkaar en gecombineerd ook nog goedkoper dan de delen apart. Is het verstandig nog verder te gaan met de integratie.

Alles in de TV

Enkele jaren geleden heb ik in Japan een paar keer enkele prototypes gezien van televisies, waar men al een huiscomputer had ingebouwd. Vooral bij MSX computers, die kwamen toch al uit dezelfde fabrieken als de TV's, zag men dat toen wel zitten. Toch heeft die combinatie het nooit gehaald, men heeft nooit een computer-TV kunnen kopen en daar zijn een aantal goede redenen voor.

Een aantal technische bezwaren, zoals het feit dat een TV-scherm in het algemeen te weinig scherpte en beeldkwaliteit biedt om goed als computermonitor te kunnen dienen, zijn ondertussen achterhaald. Juist in een combinatie, waar men de computersignalen zonder een verstorende modulatie-stap direct naar de buis kan sturen, kan men een hele acceptabele beeldkwaliteit bereiken. De moderne TV/monitor combinaties maken dit ook wel duidelijk. Wel is het zo, dat een huiskamer-TV meestal wel wat groter is dan een monitor, de beelddiagonaal van een monitor is zelden groter dan 40 cm. Dat betekent, dat men niet zo dicht bij een TV scherm kan gaan zitten als men bij een monitor gewend is. Maar ook dat is met een wat langere toetsenbordkabel, of desnoods een infrarood-koppeling tussen toetsenbord en de in de TV ingebouwde CPU, wel op te lossen.

Time-sharing is hèt probleem bij een te grote integratie van functies in één apparaat.

De TV en de computer combineren is sociaal gezien niet optimaal, omdat men een soort "time-sharing" probleem de huiskamer binnenhaalt: nèt op het moment, dat vader naar het nieuws wil kijken, zit zoonlief druk in een compileer-slag met zijn programma. Zelfs met de gewone TV en al die kabelnetten is die problematiek al overbekend en er is nauwelijks behoefte aan nog een verstorende factor erbij. De hele trend, om apparatuur te combineren tot super-éénheden, past in dit opzicht overigens beter bij de Japanse mentaliteit. Daar stopt men traditioneel zoveel mogelijk funkties bij elkaar. Wat er dan op een bepaald moment mee gedaan wordt , welk programma er gedraaid wordt of welk net ingeschakeld, dat is daar meer een groepsbeslissing, waar iedereen zich dan bij neerlegt. Dat is niet alleen bij Audio/Video zo, ook bij kantoorcomputers is iets dergelijks te zien. In Japan bouwt men vaak de computer, het beeldscherm en de printer allemaal in één kastje. Alles bij elkaar, niet omdat men iedere medewerker alle funkties wil geven, maar omdat opsplitsen minder goed past bij de Japanse werksfeer. Wanneer men in een kantoor ziet, hoe een beslissing tot stand komt, dan blijkt men vaak met alle medewerkers bij elkaar rond één computer druk discussiërend tot een gemeenschappelijke mening te komen. Is die bereikt, en dat kan soms lang duren, dan gaat de computer aan de gang, er worden reeksen uitdraaien gemaakt en ieder gaat met zijn kopie daarvan terug naar zijn eigen plek. Het idee van een gemeenschappelijke printer of het communiceren via een lokaal (computer)-netwerk past helemaal niet in die opzet. Dat zou er toe kunnen leiden, dat sommigen eigen informatie hebben, daarmee tot eigen conclusies komen, die kunnen afwijken van wat de rest van het bedrijf denkt en doet.

Een veel fundamenteler reden om de TV en de computer uit elkaar te houden is het onderscheid tussen actieve en passieve recreatie.

Terug naar de computer-TV combinatie. Daarbij gaan we er even van uit, dat de huiscomputer toch nog overwegend voor het plezier wordt gebruikt, spelletjes en zelf (leren) programmeren hebben geen direct nuttige of productieve funktie. Een computer alleen gebruiken om er naar te kijken, dat komen we niet veel tegen, de aloude kleuren-kaleidoscoop met door elkaar schuivende kleuren-plaatjes laat men echt geen dagen op het beeld aan de gang. Nee, juist door dat toetsenbord kan men aan de gang, communiceren met de CPU, zorgen dat het ding doet wat we willen of het middels programmatuur (inter)-activeren. Computeren is een activiteit, waarbij men de hersens moet gebruiken en bij spelletjes ook nog snel moet reageren. Bij de TV is dat, met uitzondering van wat educatieve toepassingen, veel minder het geval: het is een medium dat vaak zeer passief geconsumeerd wordt.
Het bij elkaar brengen van het actieve (computer) en passieve (TV) gebruik in één unit is dus een ongelijkwaardige combinatie. Het lijkt op het plaatsen van een hometrainer voor de TV en dan al fietsend kijken naar een reportage over de Tour de France.

Deze cursus heeft als doel snel te begrijpen wat machinetaal nu in feite is en wat je er in de praktijk mee kunt doen. Het belangrijkste is, dat je zelf straks in machinetaal kunt programmeren. Met hulp van deze cursus ligt dat binnen ieders bereik. In de cursus gebruiken we de 6500-familie als microproccessor, niet omdat deze instructieset eenvoudig is, maar omdat de meeste computer-bezitters een C-64 of C-128 hebben waarin een soortgelijke chip aanwezig is.

Cursus MACHINETAAL

door Tjipke van der Land

deel 4 : Getallen

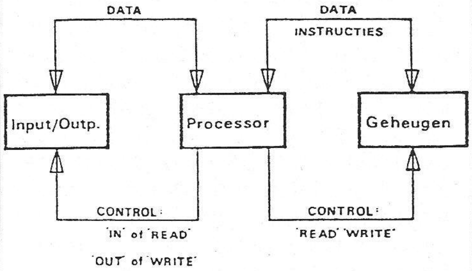
De vraag die het meest gesteld wordt is : 'Hoe snel kun je nu eigenlijk in machinetaal programmeren, hoelang duurt het voordat je wat vaardigheid hebt gekregen?'. Het antwoord op deze vraag hangt sterk af van de persoon die hem stelt. Het kan best zijn dat je al enige ervaring met je computer hebt, dan gaat het snel, maar het kan ook zijn dat je er nog helemaal niets van af weet. In het laatste geval zal het ongetwijfeld wat langer duren, voordat je de 6500-kunsten meester bent.

Om in machinetaal aan het werk te kunnen is het niet echt nodig om een meesterprogrammeur in Basic te zijn, maar wel moet je weten hoe je de computer moet bedienen. Omdat er in de vorige afleveringen nogal hard van stapel is gelopen, doen we vanaf deze les het begin nog een dunnetjes over.

De reden waarom machinetaal de voorkeur verdient boven Basic is al beantwoord in een voorgaand artikel. Hoe het precies werkt komt allemaal nog aan de orde, want het is logisch dat je dit niet in één keer kunt bevatten. Waar we mee beginnen is jammer genoeg nogal droge stof, waar je in 't begin niet veel mee kunt, omdat dit de **basis-kennis** is die je nodig hebt, om later zelf te kunnen programmeren.
Wat je moet leren is **de taal** van de microprocessor en hoe de microprocessor werkt. Dat dit gemakkelijker gezegd is dan gedaan zal wel duidelijk zijn. We zullen eerst de globale werking van de microprocessor eens wat beter bekijken.

De microprocessor

De microprocessor is het hart van de computer en kun je als een soort politieagent beschouwen die de hele werking van de computer op zich neemt. De processor is echter niet meer dan een **onderdeel** van de computer en je kunt beslist niet stellen dat een computer alleen uit een microprocessor bestaat. Een computer bevat als hoofdonderdelen in principe: een microprocessor, lees- en schrijf-geheugen, een programma, *'waar hij mee werkt* en de in- en uitvoer voor bijvoorbeeld toetsenbord of videoscherm. Technisch gesteld komt dit neer op een microprocessor , RAM (Random Access Memory), ROM (Read Only Memory) en I/O (input/output). Dit zijn de onderdelen die samen de computer vormen.



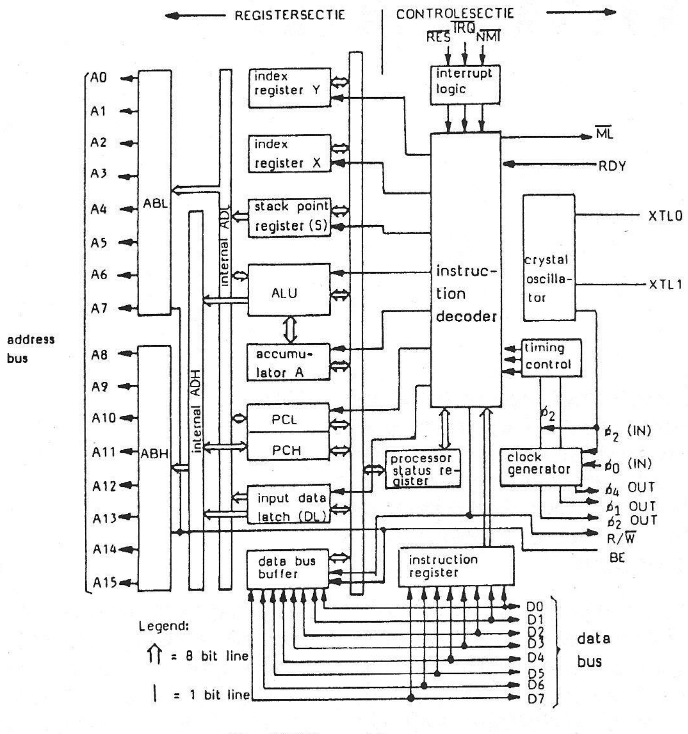
Computer-schema

Een volgende stap is het analiseren van elk blokje van het computerschema, en we beginnen met het belangrijkste onderdeel, de microprocessor. Het is niet de bedoeling om alle mogelijkheden en technische specificaties te behandelen, want dan zijn we vijf afleveringen verder nog niet aan de 6500-instructieset toe. Voordat je met het programmeren kunt beginnen is een beetje kennis van de microprocessor toch wel nodig. Als je wilt gaan autorijden, moet je toch ook weten hoe een auto werkt, nietwaar. Verder op vind je het complete schema van de 6502 microprocessor, waarvan de werking in de loop van deze cursus zal worden uitgelegd.

Overzicht

Waar het op dit moment om gaat is een globaal overzicht van de werking van de computer. De processor op zichzelf is niet zo'n intelligent onderdeel. Het enige wat hij doet is het domweg uitvoeren van **instructies** die hij opgedragen krijgt. Wat hij natuurlijk weet, is de betekenis van de instructies die worden aangeboden. Binnen in de microprocessor is net genoeg intelligentie aan-

wezig voor de besturing en de ontleding van instructies en data. Dit noemt men het **microprogramma** in de processor, wat hardwarematig bekeken ongeveer 2/3 deel in beslag neemt van het bijna niet te onderscheiden vierkante gedeelte binnen in de zwarte behuizing van de microchip. Het overgebleven deel is voor de echte hardware, dus de duizenden transistoren en weerstanden die in de microprocessor zitten verstopt.



De 6502 architectuur

Bussen

De 6502 microprocessor heeft 40 aansluitpennen, en de aansluitpennen die voor ons van belang zijn, zijn de **data-** en de **adresbus** die samen al 24 aansluitpennen reserveren, 16 voor de adresbus en 8 voor de databus.

De **databus** is bi-directioneel, beschikt dus over mogelijkheden in twee richtingen gegevens te versturen, - dus kan data in- en uitvoeren-, en wordt gebruikt om data (informatie) over te brengen van en naar de microprocessor. Deze databus is 8 bits breed, wat eigenlijk betekent, dat de informatie over acht draadjes wordt getransporteerd. Deze data bestaat uit zowel **instructies** voor de processor als **gegevens** die gebruikt worden voor bijvoorbeeld berekeningen.

De **adresbus** van de microprocessor is 16 bits breed, (dus 16 draadjes) en is bedoeld -zoals de naam al doet vermoeden- voor het **adresseren**. Dit adresseren gebeurt alleen door de processor, daarom is de adresbus **niet** bidirectioneel. Het is net als het versturen van een brief naar een adres. Het is niet de bedoeling dat de brief terugkeert naar de afzender, vandaar het eenrichtingsverkeer.

Getallen

De vraag naar de hoeveelheid adressen die door de **adresbus** kunnen worden aangesproken is heel eenvoudig te berekenen, namelijk **2 tot de 16e macht**. De 2 staat in dit geval voor de twee signaalmogelijkheden van een adreslijn, **0 en 1**, en de **16** voor het aantal adreslijnen. Na berekening van $\mathbf{2^{16}}$ komt je op **65536**. Dit is een getal wat je verder weinig in de computer-literatuur tegen zult komen, omdat in de computer-techniek en -verwerking een beetje anders tegen het getal 1000 wordt aangekeken. Ga je vanuit het binaire talstelsel **machtsverheffen** dan is het dichtsbijzijnde getal bij de 1000 het getal **1024**. Omdat je binair niet precies op 1000 of 1K komt wordt 1024 voor het gemak gelijk gesteld aan 1000. Deel je nu 65536 op 1024 dan blijkt, dat er 64 uitkomt en dat refereert al snel naar de welbekende 64 Kbyte computer, zoals de C-64. Dus als we in het vervolg over 1 Kbyte (Kilobyte) praten, bedoelen we eigenlijk 1024 bytes.

Niet alle **64K** adressen zijn voor het gebruikers-geheugen bedoeld, er zijn ook nog andere onderdelen in een computer die geadresseerd moeten worden, bijvoorbeeld de delen die het toetsenbord, het videoscherm en de printer besturen. De 6500 serie processoren zijn **'memory-mapped'**. Dit betekent dat zowel de I/O (input/output), als het geheugen in hetzelfde adresgebied wordt aangestuurd, dus niet met een aparte I/O databus, zoals bij de Motorola 8080 processoren. Hieronder staat een voorbeeld van een memory map van een C-64, en hieruit kun je al opmaken dat niet de hele adresruimte wordt gebruikt voor het RAM-geheugen, maar ook voor I/O, video, ROM-pack, en kernal ROM.

Opdrachten

Voordat een microprocessor aan het werk kan, moet je hem precies vertellen wat hij moet gaan doen. Deze opdrachten moeten in een serie hele kleine stapjes aan de processor worden verstrekt. Complexe opdrachten, zoals we aan mensen zouden kunnen geven, kan de microprocessor niet verwerken. In plaats van '*Ga naar de deur*' zullen we tegen de 6510 moeten zeggen '*zet het rechter been voor het linker om te gaan lopen*' etcetera. Op dit **basisniveau** moet je de microprocessor stap voor stap vertellen wat hij moet gaan doen.

Voorbeeld

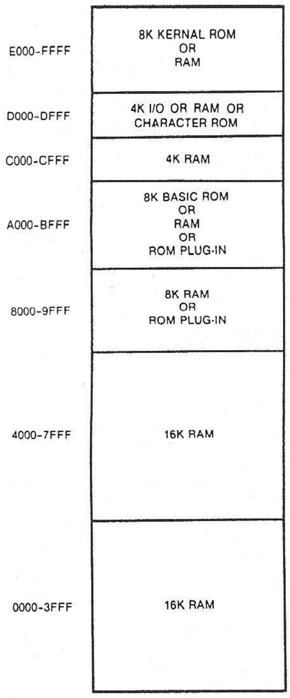
We nemen als voorbeeld het optellen van twee getallen die ergens willekeurig in het computer-geheugen staan, en we gaan dit stap voor stap doornemen op dezelfde manier als de microprocessor dit verwerkt. Het **optellen** van twee getallen lijkt naar menselijke begrippen één instructie, maar de microprocessor gebruikt veel meer voor het simpel optellen van deze twee getallen.

Instructie 1:
Maak het reken-register schoon

Instructie 2:
Laad rekenregister met getal 1 uit het geheugen

Instructie 3:
Tel getal 2 uit het geheugen op bij getal 1 en plaats het opnieuw in het rekenregister.

Zo telt de computer twee simpele getallen op, dus moet je je maar eens voorstellen in hoeveel fasen een vermenigvuldiging of een ander soort berekening gedaan wordt.



Geheugen indeling C-64

Talstelsels

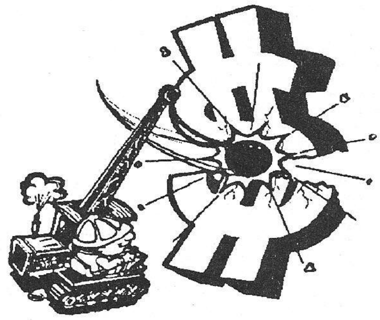
De microprocessor werkt met het binaire stelsel, en **'bi'** staat voor twee. De twee mogelijkheden van dit stelsel zijn **0 en 1**. Dit zijn dan ook de enige mogelijkheden die voorkomen op de data- en de adresbus. De letter **'A'**, die je op het scherm ziet, bestaat eigenlijk uit de binaire code **0100 0001**. Deze code is 8 cijfers (digits) lang, we noemen dat *8 bits*. Zo'n 'formatie' cijfers heet ook wel een **binair woord**. Je kunt uit de breedte (8) van het woord opmaken dat hij over de databus is getransporteerd, wat de computerlogica gelukkig ten goede komt, al zal zeker niet iedereen dit maar zo begrijpen.
We zullen nu eens bekijken hoe je kunt **omrekenen** van binair naar hexadecimaal en vervolgens naar decimaal.
Het is heel erg belangrijk dat je dit kunt, want je krijgt in het verloop van deze cursus veel te maken met hexadecimale getallen die nogal eens omgerekend moeten worden naar binair of decimaal.

Hexadecimale getallen gebruiken we eigenlijk om de binaire getallen beter leesbaar te maken, want het is veel gemakkelijker om **05** te lezen dan **0101** of **3C** in plaats van **0011 1100**. Het woord **hex** staat voor **zes** en na samenvoegen van hex(a) en decimaal blijkt dit 6+10=16 te zijn. In de digitale techniek kunnen we helaas niet straffeloos het decimale (10-tallig) stelsel toepassen in plaats van het hexadecimale, omdat een veelvoud van het binaire stelsel zelden overeenkomt met het decimale stelsel. In het hexadecimale stelsel blijkt dit veel beter te passen. Om dit toe te lichten staat hieronder een tabel afgebeeld waar dit in terug te vinden is.

TABEL

DEC	HEX	BINAIR
0	0	0 0 0 0
1	1	0 0 0 1
2	2	0 0 1 0
3	3	0 0 1 1
4	4	0 1 0 0
5	5	0 1 0 1
6	6	0 1 1 0
7	7	0 1 1 1
8	8	1 0 0 0
9	9	1 0 0 1
10	A	1 0 1 0
11	B	1 0 1 1
12	C	1 1 0 0
13	D	1 1 0 1
14	E	1 1 1 0
15	F	1 1 1 1

Hier kun je zien dat het hexadecimale (16) stelsel doortelt tot de letter **F**, waar het decimale (10) stelsel maar tot 10 kan tellen. Omdat je toch een notatie moet hebben, die beter te gebruiken valt worden de binaire enen en nullen meestal hexadecimaal verrekend. Het nut hiervan is



Hexadecimalen

duidelijk, want hierdoor worden 4 binaire digits omgezet naar 1 hexadecimaal getal. In het decimale stelsel is dit niet mogelijk, omdat je geen enkele logische verhouding kunt creëren tussen binaire en decimale getallen. Hexadecimaal is dit wel het geval, want 2 tot de vierde macht (2^4) is 16 (decimaal), dus 0F hexadecimaal.
Alle notaties bij machinetaal in de vorm van een getal of tekst worden dan ook hexadecimaal geschreven. Het kenmerk, waaraan je kunt zien, met een hexadecimaal getal te doen te hebben, maak je duidelijk door een **$** teken voor het getal te plaatsen. Bij binair doe je hetzelfde met een **%** teken.
Dus 11 = $0B = %1011.

Omrekenen

We zullen nu een poging wagen om een getal van het ene stelsel naar het andere om te rekenen. Bij erg kleine getallen geeft dit natuurlijk geen problemen, want die zijn zo af te lezen uit de tabel, elders in dit artikel. Maar voor grotere getallen komt er wat meer kijken. We nemen daarvoor het getal **123** decimaal. Dit getal kan worden voorgesteld als:

100 x10	10 x10	1 (0 x10)
1	2	3
x---	x---	x---
100	20	3 (= 123)

Na splitsing in grondgetallen kun je zien dat 1 een honderdtal is, 2 een tiental en 3 een eenheid. Naarmate een getal groter wordt, wordt de waarde met 10 vermenigvuldigd; dus 1,10,100, 1000,10000 enzovoort.

Ditzelfde geldt ook voor andere stelsels, allemaal uitgaande van hun grondtal.
In het hexadecimale stelsel hebben we in plaats van 10 dus 16 als grondtal.
We gaan hier als voorbeeld omrekenen van hexadecimaal naar decimaal. We nemen daarvoor **$123** als hexadecimaal getal, en na splitsing ziet dit er zo uit:

256 x16	16 x16	1 (0x 16)
1	2	3
x---	x---	x---
256	32	3

De uitkomst is dus 256+32+3 =291
Nog een voorbeeld met als hexadecimaal getal **$C3F**, wat we naar decimalen zullen terug zetten :

256 x16	16 x16	1 (0x16)
C	3	F
x---	x---	x---
3072	48	15

De uitkomst van deze berekening krijgen we, door de getallen 3072, 48 en 15 bij elkaar te tellen. Het wordt dus in decimalen 3135.
Bij dit stelsel kun je duidelijk zien, dat elk getal wat er bij komt met grondtal 16 wordt vermenigvuldigd, dus 1,16, 256, 4096, 65536

En dan nu een voorbeeld van het omrekenen van **binair** naar **decimaal**. Ook dit gaat op dezelfde manier als bij hexadecimaal en decimaal. We nemen als voorbeeld **%01111011** :

```
128 64 32 16 8 4 2 1
--------------------------------
  0  1  1  1 1 0 1 1
x--- x--- x--- x--- x--- x--- x--- x---
  0 64  32 16 8 0 2 1
```

De uitkomst hiervan wordt opgeteld 123. Je kunt dit omrekenen gelukkig ook sneller doen, door het binaire woord op te splitsen in groepjes van vier digits, en daarna te vergelijken met de **decimaal /hexadecimaal /binair** tabel om hiermee per groepje een hexadecimaal getal te krijgen bijvoorbeeld:

```
0111    1011
----    ----
  7       B
```

Op deze manier kun je veel sneller van binair, via hexadecimaal, naar decimaal omrekenen, dan met de rechtstreekse methode het geval is.

Dit waren een paar voorbeelden van het rekenen van binair en hexadecimaal naar decimaal. Met deze kennis ben je er nog niet, want je

De hexadecimale getallen worden altijd aangeduid door een dollarteken voor het getal te plaatsen.

Bij binaire getallen maken we gebruik van het procent-teken.

moet ook van decimalen terug kunnen rekenen naar binaire of hexadecimale getallen. Als voorbeeld rekenen we het getal 123 terug naar hexadecimaal. Het gemakkelijkste is het delen van dit getal op de hoogst mogelijke macht van 16, je kunt daarbij dus kiezen uit 1, 16, 256 en 4096, zodat het een getal oplevert groter dan 1, daar bedoel ik mee dat het delen van 123 op 256 een getal oplevert, dat **kleiner** is dan 1, waar in dit geval niet met succes een uitkomst uit is te toveren. Wat wel werkt is het delen van dit getal op 16, en het verschil weer delen op het getal wat daar onder ligt, voorzover dat mogelijk is. Een voorbeeld zal hier wel wat duidelijkheid verschaffen:

123/16 = 7 + rest. Het getal 16 kan dus 7 keer uit 123,
7 x 16 = 112 , waarbij er
123 - 112 = 11 als rest overblijft. **11** wordt hexadecimaal geschreven als **$B**, waardoor de uitkomst **$7B** wordt.

Dit zelfde is ook te verwezenlijken bij het overzetten van decimale getallen naar het binaire stelsel, met weer als voorbeeld het getal **123**:
Nu moet je gebruik maken van de hoogste macht van het getal 2, dus 1, 2, 4, 8, 16, 32, 64, 128, 256 enz.

123 / 64 = 1
Het getal 64 kan 1 keer uit 123,
64 x 1 = 64
en mag nooit meer zijn dan 1
123 - 64 = 59
Er blijft 59 over.

Dan vervolgen we de gang van de berekening hieronder zonder verder commentaar

```
-----------------
59 / 32   = 1
32 x 1    = 32
59 - 32   = 27
-----------------
27 / 16   = 1
16 x 1    = 16
27 - 16   = 11
-----------------
11 / 8    = 1
8 x 1     = 8
11 - 8    = 3
-----------------
3 / 4     = 0
4 x 0     = 0
3 - 0     = 3
-----------------
3 / 2     = 1
2 x 1     = 2
3 - 2     = 1
-----------------
1 / 1     = 1
1 x 1     = 1
1 - 1     = 0
-----------------
```

Vervolgens kunnen we nu aan de hand van bovenstaande berekeningen de tabel samenstellen:

```
64 32 16 8 4 2 1
--------------------------
 1  1  1 0 1 1 1
```

Op deze manier kun je weer terug rekenen van decimale naar binaire getallen.
Wat je ook kunt doen, is het rekenen van decimaal naar hexadecimaal, en vervolgens aan de hand van de conversie-tabel bekijken welke binaire code er bij hoort.

We hebben nu nog even snel geroken aan het rekenen, en berekenen van de diverse talstelsels die in de machinetaal nogal vaak voorkomen. De bedoeling van dit



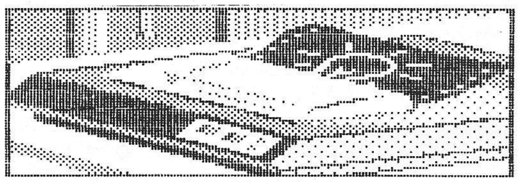
alles is natuurlijk, om het werken met de diverse stelsel wat te verduidelijken. Eerder in deze cursus hebben we het er ook al over gehad, maar gezien de hoeveelheid reacties van lezers die het allemaal nogal ingewikkeld vonden hebben we in deze les de getallen nog maar eens bij de kop genomen. Nu moet je niet direct de moed laten zakken, als je het niet meteen begrijpt. Een goede oefening is het opschrijven van een complete tabel, zovel hexadecimaal, decimaal als binair, van alle getallen tussen 0 en honderd. Als je die allemaal op een rij hebt staan - zelfgeschreven-krijg je een goed overzicht van de mogelijkheden die deze stelsels bevatten. Tien tegen één, dat ze daarna ook geen geheimen meer voor je hebben.

De volgende keer gaan we verder met de bespreking van een belangrijk onderdeel van de computer, het geheugen. Voor nu, veel sterkte en ik zou zeggen: 'Reken maar raak'.

Tjipke van der Land.

Desktop Publishing wint steeds meer aan populariteit in de PC-wereld. In dit artikel gaat Peter Boncz in op de vraag of dit ook bij de 'goeie ouwe' C-64 zo is en wat de gewone gebruiker nu precies aanmoet met Desktop Publishing.

Desktop Publishing op de C-64



Desktop Publishing, het intikken, opmaken en uiteindelijk afdrukken van pagina's tekst en illustraties, is in de PC-wereld een ware rage. Ook de C-64 is hier niet aan voorbij gegaan. Nu de eerste stormen wat geluwd zijn is het nuttig om de zaken eens op een rijtje te zetten. Daartoe namen we een greep uit de Desktop Publishing pakketten. We bekeken The Newsroom van Springboard, The Printshop van Broderbund, The Printmaster van Unison World (via Berkely Softworks) en GEOS, eveneens van Berkeley Softworks.

Het betreft hier dus allemaal Desktop Publishing programma's, d.w.z. programma's waarmee tekst ingevoerd kan worden, gekozen kan worden uit verschillende lettertypes, de layout op de computer kan worden bepaald en eventueel plaatjes kunnen worden toegevoegd. Bovendien werken ze allemaal volgens het WYSIWYG principe. Dit houdt in dat de printer precies afdrukt, wat er op de monitor weergegeven wordt.



Prijzig

In het bedrijfsleven is Desktop Publishing een veel voorkomende manier van automatiseren. Allerlei circulaires, krantjes, e.d. kunnen nu door de gebruiker zelf, zonder de (dure) zetter, gedrukt worden. Men maakt dan meestal gebruik van een snelle PC (een AT) met hard disk, gekoppeld aan een laserprinter met prijskaartje van *f* 15.000,- toch wel een dure configuratie.
Dat het ook met minder kan wordt door programma's als **The Printmaster** bewezen: voor een paar tientjes kan de C-64 gebruiker zelf aan de slag gaan. Natuurlijk zal de kwaliteit van de afdrukjes sterk van het soort printer afhangen, en daarom zullen de resultaten van hobbyisten met eenvoudige dot-matrixprinters wat slechter zijn dan die van bovengenoemde professionals. Maar over het algemeen maakt dat niet veel uit: als hobbyist stelt men ook minder hoge eisen, en de kwaliteit is zeker wel acceptabel te noemen.
Om de eigenschappen van alle programma's goed naar voren te laten komen, geven we nu een beschrijving van de mogelijkheden en voor- en nadelen van de geteste programma's.

The Newsroom

Toen het Amerikaanse softwarehuis Springboard in 1984 het DeskTop Publishing programma **The Newsroom** op de markt bracht kon zij nauwelijks vermoeden welk succes dit pakket op scholen, bij hobbyisten en zakelijke gebruikers zou hebben. Springboard werd in 1982 opgericht door de voormalige leraar John Pauelson. Hij hield zich bezig met het schrijven van kwalitatief goede educatieve software, en met succes: een van zijn eerste programma's, **Early Games for Young Children**, werd direct een grote topper en stond 99 weken in de Softsel Hot List in Amerika. The Newsroom heeft dit succes inmiddels wel geëvenaard. In 1986 werden alleen al in de VS 300.000 exemplaren verkocht. Het programma is bedoeld om thuis met behulp van een C-64 geïllustreerde kranten, folders, nieuwsbrieven en reklamebulletins te kunnen ontwerpen en op eigen printer af te drukken.
Met The Newsroom kan de gebruiker krantekoppen, banieren en 5 verschillende letterfonts (lettertypen) met illustraties combineren. Deze illustraties kan de thuisuitgever halen uit de 600 graphics bevattende Art Gallery op de zgn. Clip Art disks. Inmiddels zijn daar nu Clip Art Volume 2 (met 800 'business' plaatjes) en Clip Art Volume 3 bijgekomen. Er zijn bovendien 'graphics expanders' in omloop, waarmee men een stuk HIRES geheugen als clipart foto kan saven en in een zelfontworpen krantje gebruiken.

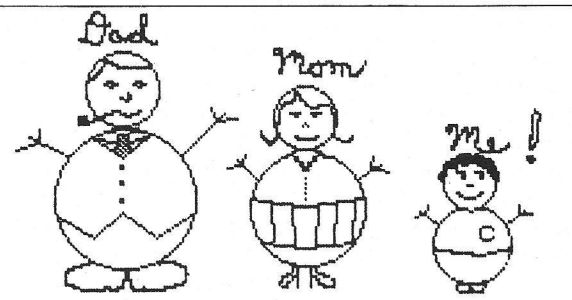


The Newsroom is verdeeld in verschillende delen:
- **copy desk**. Hierin kan men tekst intypen in blokken. Met deze blokken kan later in Layout geschoven worden.
- **layout**. Het bepalen van de opmaak van de krantepagina. Een pagina is opgebouwd uit foto's, de krantekop

(banner) en de in copy desk ingevoerde tekst-blokken.
- **wire service** is een modem-programma, waarmee de thuisuitgever gegevens met andere Newsroom gebruikers kan uitwisselen.
- **press** omvat het printer instel en afdruk menu.
- **photo lab** stelt de gebruiker in staat ClipArt graphics te wijzigen. Men heeft daartoe uitgebreide drawing opties, zoals line, block, fill, pattern, verschillende tekenpennen en 'OOPS' voor het herstellen van gemaakte fouten.
- **banner.** Hier wordt de krantekop gemaakt.

The Printshop en The Printmaster

Dit zijn twee programma's die eigenlijk als twee druppels water op elkaar lijken. Daarom nemen we ze in deze bespreking samen. The Printshop is gemaakt door Broderbund Software en de Printmaster door Berkeley Softworks. Beide programma's zijn bedoeld voor het maken van felicitatiekaarten, affiches, wimpels etc.



Johnny's first art masterpiece. The Coyle family snowmen.

The Printshop is verreweg het oudste Desktop Publishing programma voor de C-64. Met dit programma brengt u de drukkerij thuis. Het eerste wat men doet, na het laden van het programma, is het kiezen van de 'set-up' optie. Dit doet men door de cursor over het zestal mogelijkheden te bewegen. Ernaast wordt dan een grafische afbeelding van de optie gegeven. (Bij 'set-up' wordt bijvoorbeeld een printer afgebeeld.) Men kan dan kiezen tussen verschillende printerdrivers (aanstuurprogramma's), voor zowel paralelle als seriële printers. Ook kan de printer van tevoren getest worden.

De 5 andere opties naast SET-UP zijn:
1. Felicitatiekaarten
2. Briefpapier
3. Affiches
4. Koppen, Wimpels (banners)
5. Toverscherm. De computer ontwerpt hier zelf allerlei tekeningen, motieven, e.d. om een ingetikt stukje tekst.

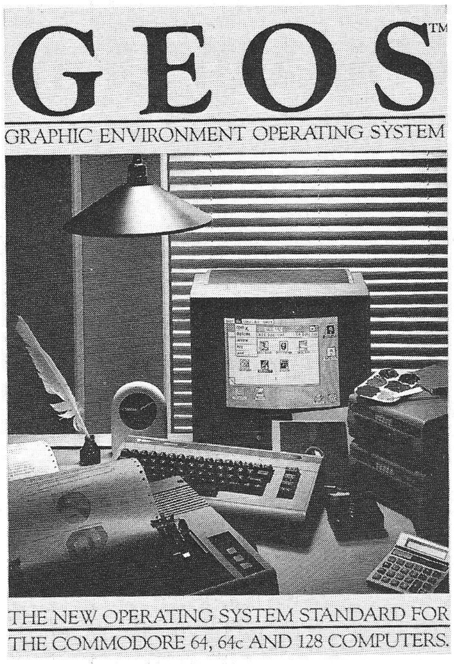
In The Printmaster is het originele 'toverscherm' van The Printshop vervangen door de kalender. Een nadeel van The Printshop is weer, dat de preview functie ontbreekt. Men kan dus vóór het uitprinten niet zien, wat het resultaat worden zal. The Printmaster bezit wel zo'n Preview functie. Bij alle bovengenoemde mogenlijkheden kan er gekozen worden uit 8 verschillende "fonts", 9 verschillende randversieringen, 50 voorwerpen en 10 dessins.

Het prijsverschil tussen beide programma's is aanzienlijk: The Printmaster kost *f* 100,- en The Printshop *f* 200,-. En dit, terwijl The Printshop nauwelijks meer presteert dan The

Printmaster. (The Printshop heeft het magic screen, en er worden briefpapier en enveloppen bijgeleverd.)

GEOS

GEOS is het nieuwste DTP-pakket dat voor de C-64 verschenen is. Naast de Desktop Publishing kant van dit fabuleuze stuk software, heeft GEOS natuurlijk nog veel meer revolutionaire

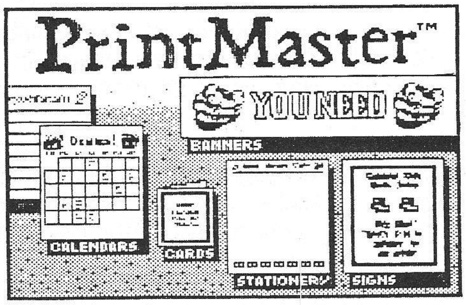


eigenschappen, waar we in dit artikel niet op in kunnen gaan. We zullen ons dus tot de WYSIWYG eigenschappen van **Geowrite** (de tekstverwerker in het GEOS pakket) en in minder mate **Geopaint** (tekenprogramma) bezighouden.
Geowrite is eigenlijk de eerste WYSIWYG tekstverwerker voor de C-64. Het is dus een tekstverwerker waarbij de gebruiker verschillende fonts kan gebruiken, verschillende stijlen heeft, plaatjes kan inlassen, en zelf de layout kan bepalen. Geowrite op zich kan niet veel meer, maar er zijn een aantal uitbreidingen voor deze tekstverwerker gekomen. Ten eerste is er het fontpack met 20 extra fonts. Ten tweede is er het **Deskpack**, dat connecties kan leggen met de GEOS database **Geodex**. Er kan met GEOS dus ook aan mailmerge gedaan worden.

Het intypen van tekst in Geowrite verliep vooral in de eerste versies erg langzaam. Naast de lange wachttijden voor aangeroepen fonts, was vooral het aanwijzen met de joystick van bijv. een verandering in stijl een langdurige aangelegenheid. Met de nieuwe uitbreidingen en updates is het mogelijk dit probleem te omzeilen: commando's kunnen ook met toetsen worden aangeroepen.Andere nadelen zoals de traagheid en het ontbreken van de mogelijkheid in kolommen te werken, bleven ook in de latere versies. Kranten kunnen er dus niet mee gemaakt worden, de doelgroep van Geowrite omvat de meer veeleisende brievenof werkstukkenschrijvers.
Gezien de populariteit van deze computertoepassing was het te verwachten dat er meer DTP programma's kwamen. En dat is ook gebeurd: Springboard kondigde **The Newsroom Pro** en **Certificate Maker** (voor het maken van allerlei certificaten en diploma's) aan en Unison World komt met **The Printmaster Plus** en **The Newsmaster**, een regelrechte concurrent van The Newsroom. GEOS, tenslotte, is nog volop in ontwikkeling en uitbreiding.

Conclusie

Desktop Publishing programma's zijn er in soorten en maten, het is voor elk wat wils. Mensen die een wijk- of schoolkrant willen maken zijn beter af met The Newsroom, en degenen die hun C-64 voor het maken van kaartjes willen gebruiken kunnen beter The



Printshop of The Printmaster aanschaffen. Overigens is het zo dat alle programma's zo groot zijn, dat ze alleen op disk uitgebracht worden. Als we naar de prijs kijken is GEOS, met de meeste mogelijkheden, een van de duurste (*f* 180,-) (en dat is dan nog zonder Fontpack of Deskpack!), en The Newsroom is met *f* 150,- ook niet goedkoop. The Printmaster is, hoewel iets minder uitgebreid, naar verhouding veel goedkoper (*f* 99,-) dan de vrijwel evenveel presterende Printshop (*f* 199,-). Maar het laatste woord is op dit gebied natuurlijk nog lang niet gesproken. ●

P. Boncz

MI$_S$SERS

We hebben in onze vorige uitgave wat problemen gehad met de listings van de C-16. **Tempo typen 2** hebben we daarom in het listinggedeelte van dit nummer in zijn geheel herhaald.
In **Schuifspel** moet u de volgende veranderingen aanbrengen:

```
20 color4, 2:color15, 2:color0,
   2:print"[clr-home]"
50 fora=1 to 4:forb = 1 to 4: char,
   1+a*4, b*4, s$(v(a,b)) : nextb :
   nexta
54 get key Q$:az=az + 1:char,
   17,23,str$(az) : if q$=chr$ (17)
   then z = 1 : goto 60
```

In regel 1040 moet u het woord collect vervangen door dec.
De regels: 52, 120, 140, 160, 180, 220, 240, 260, 280, 320, 340, 360, 380, 420, 440, 460, 480, 1390, 1400, 1410 en 1420 moeten beginnen met char.